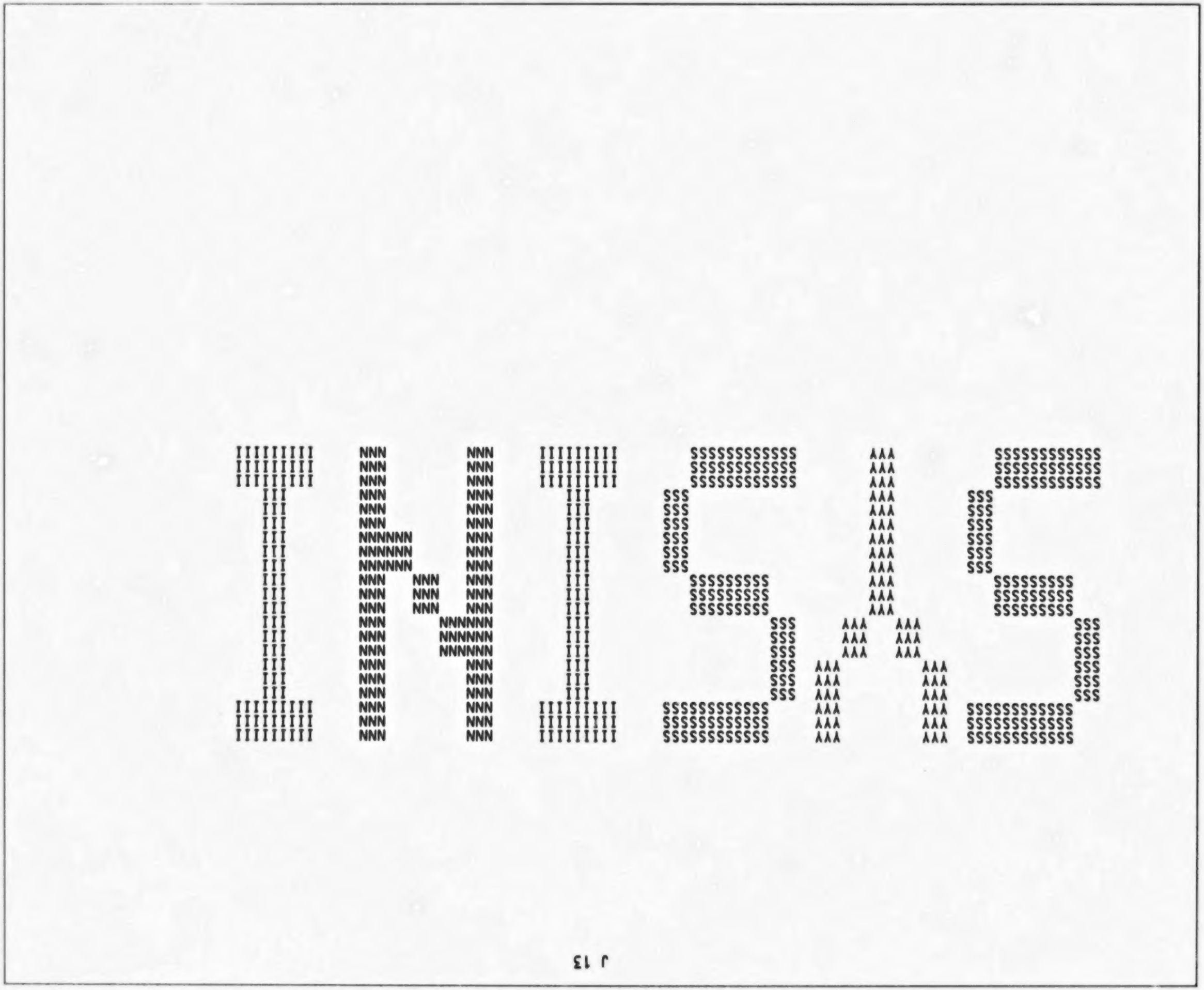
IIIIIIIII NNN NNN IIIIIIIII SSSSSSSSSSS AAA SSSSSSSSSSS
IIIIIIIII NNN NNN IIIIIIIII SSSSSSSSSSS AAA SSSSSSSSSSS
IIIIIIIII NNN NNN IIIIIIIII SSSSSSSSSSS AAA SSSSSSSSSSS
III NNN NNN III SSS AAA SSS
III NNN NNN III SSS AAA SSS
III NNN NNN III SSS AAA SSS
III NNNNNN NNN III SSS AAA SSS
III NNNNNN NNN III SSS AAA SSS
III NNNNNN NNN III SSS AAA SSS
III NNN NNN NNN III SSSSSSSSS AAA SSSSSSSS
III NNN NNN NNN III SSSSSSSSS AAA SSSSSSSS
III NNN NNN NNN III SSSSSSSSS AAA SSSSSSSS
III NNN NNNNNN III SSS AAA AAA SSS
III NNN NNNNNN III SSS AAA AAA SSS
III NNN NNNNNN III SSS AAA AAA SSS
III NNN NNN III SSS AAA AAA SSS
III NNN NNN III SSS AAA AAA SSS
III NNN NNN III SSS AAA AAA SSS
IIIIIIIII NNN NNN IIIIIIIII SSSSSSSSSSS AAA AAA SSSSSSSSSSS
IIIIIIIII NNN NNN IIIIIIIII SSSSSSSSSSS AAA AAA SSSSSSSSSSS
IIIIIIIII NNN NNN IIIIIIIII SSSSSSSSSSS AAA AAA SSSSSSSSSSS

```
  SSSSSSSS  YY      YY   SSSSSSSS    IIIIII    NN      NN    IIIIII    TTTTTTTTTT
  SSSSSSSS  YY      YY   SSSSSSSS    IIIIII    NN      NN    IIIIII    TTTTTTTTTT
SS          YY      YY  SS             II     NN      NN      II          TT
SS          YY      YY  SS             II     NN      NN      II          TT
SS            YY  YY    SS             II     NNNN    NN      II          TT
SS            YY  YY    SS             II     NNNN    NN      II          TT
  SSSSSS        YY        SSSSSS       II     NN  NN  NN      II          TT
  SSSSSS        YY        SSSSSS       II     NN  NN  NN      II          TT
        SS      YY              SS     II     NN    NNNN      II          TT
        SS      YY              SS     II     NN    NNNN      II          TT
        SS      YY              SS     II     NN      NN      II          TT        ....
        SS      YY              SS     II     NN      NN      II          TT        ....
SSSSSSSS        YY      SSSSSSSS     IIIIII    NN      NN    IIIIII      TT        ....
SSSSSSSS        YY      SSSSSSSS     IIIIII    NN      NN    IIIIII      TT        ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

(2) 172 DECLARATIONS
(3) 490 Data Used by $ENQW Request
(4) 521 Data Used To Create Stand-Alone Configure Process
(5) 577 Data Used For Quorum disk
(6) 611 IMPURE DATA FOR $CRELNM AND $TRNLNM CALLS
(7) 672 PURE DATA FOR $CRELNM AND $TRNLNM CALLS
(8) 760 SYSTEM INITIALIZATION PROCESS
(9) 1193 SIP_GET_SYSID_LOCK - Obtain Lock for System ID
(10) 1272 SIP_CLUSTER_INIT - Cluster related initialization
(11) 1421 SIP_LOOKUP_QFILE - Perform quorum file lookup
(12) 1537 SIP_START_QUORUM_TIMER - Start the quorum disk timer
(14) 1602 SIP_MAPXQP - Create global sections for XQP
(15) 1659 SIP_IMAGE_ATT - Read header, get image attributes
(16) 1702 BOO$IMAGE_ATT - Get image attributes from image header
(17) 1749 SYSTEM INITIALIZATION KERNEL LEVEL
(18) 1821 SIP_INITPAGFIL Initialize PAGEFILE.SYS
(19) 1978 CHECK CACHE
(20) 2009 SIP_INITSWPFIL Initialize SWAPFILE.SYS
(21) 2041 SIP_INITRMS - Install RMS Image
(22) 2100 RESTORE ERROR LOG BUFFERS
(23) 2172 QIO_RWVB - Read or Write Virtual Block
(24) 2261 QIO_RWLB - Read or Write Logical Block
(25) 2335 SIP_INIWCB - ALLOCATE AND INIT A WINDOW CONTROL BLOCK
(26) 2370 ALLOCATE NON-PAGED DYNAMIC MEMORY
(26) 2409 DEALLOCATE FIL$OPENFILE CACHE
(27) 2424 SIP ERROR/MESSAGE OUTPUT
(28) 2501 SIP_SETTIME - SET SYSTEM TIME TO CORRECT VALUE AT STARTUP

```
0000     1          .TITLE  SYSINIT - SYSTEM INITIALIZATION PROCESS
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
0000     9 ;*                                                                           *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    15 ;*  TRANSFERRED.                                                             *
0000    16 ;*                                                                           *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    19 ;*  CORPORATION.                                                             *
0000    20 ;*                                                                           *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23 ;*                                                                           *
0000    24 ;*                                                                           *
0000    25 ;*****************************************************************************
0000    26
0000    27 ;++
0000    28 ; FACILITY:     SYSTEM INITIALIZATION
0000    29 ;
0000    30 ; ABSTRACT:     PERFORMS OPERATIONS NECESSARY TO GET
0000    31 ;               THE SYSTEM TO A POINT THAT IT CAN
0000    32 ;               SUPPORT ITSELF.
0000    33 ;
0000    34 ; ENVIRONMENT:  OPERATES WITHIN THE LIMITED CAPABILITIES
0000    35 ;               THE BOOT STRAPPED OPERATING SYSTEM.
0000    36 ;
0000    37 ; AUTHOR:  W.H.BROWN, CREATION DATE:  6-JAN-77
0000    38 ;
0000    39 ; MODIFIED BY:
0000    40 ;
0000    41 ;       V03-033 HH0052          Hai Huang                   28-Aug-1984
0000    42 ;               Correctly bias the reference count for the system device.
0000    43 ;
0000    44 ;       V03-032 RAS0304         Ron Schaefer                4-May-1984
0000    45 ;               Re-define SYS$SYSDEVICE and SYS$DISK so that the
0000    46 ;               correct allocation-class is available to define the name.
0000    47 ;
0000    48 ;       V03-031 CDS0001         Christian D. Saether        1-May-1984
0000    49 ;               Set the device characteristic CLU before mounting the
0000    50 ;               system disk if we intend to be a cluster.
0000    51 ;
0000    52 ;       V03-030 TMK0002         Todd M. Katz                28-Apr-1984
0000    53 ;               Completely redo how the system logical names are created.
0000    54 ;               I have done this to eliminate the last vestiges of the old
0000    55 ;               logical name system services and to optimize this code in
0000    56 ;               the process.
0000    57 ;
```

```
0000     58 ;       V03-029 DWT0212         David W. Thiel          09-Apr-1984
0000     59 ;               Add call to CNX$DISK_CHANGE when CLUB$T_QDNAME is
0000     60 ;               filled in.
0000     61 ;
0000     62 ;       V03-028 WMC0022         Wayne Cardoza           02-Apr-1984
0000     63 ;               Use XQP_RESIDENT SYSGEN parameter.
0000     64 ;
0000     65 ;       V03-027 RSH0120         R. Scott Hanna          19-Mar-1984
0000     66 ;               Make changes to SIP_LOOKUP_QFILE due to new quorum
0000     67 ;               file algorithm. Add SIP_START_QUORUM_TIMER.
0000     68 ;
0000     69 ;       V03-026 WMC0021         Wayne Cardoza           14-Mar-1984
0000     70 ;               Don't give message for NOSUCHFILE errors.
0000     71 ;
0000     72 ;       V03-025 WMC0020         Wayne Cardoza           10-Mar-1984
0000     73 ;               Make XQP a resident global section.
0000     74 ;
0000     75 ;       V03-024 ACG0399         Andrew C. Goldstein,    27-Feb-1984  12:33
0000     76 ;               Rename EXE$LOCK_DEV to IOC$LOCK_DEV
0000     77 ;
0000     78 ;       V03-023 WHM0001         Bill Matthews           17-Jan-1984
0000     79 ;               Add definition of SYS$SYSROOT and SYS$COMMON. Convert
0000     80 ;               CRELOG'S and TRNLOG to the LNM form.
0000     81 ;
0000     82 ;       V03-022 WMC0019         Wayne Cardoza           12-Jan-1984
0000     83 ;               XQP now has DZRO space, no CRF allowed.
0000     84 ;
0000     85 ;       V03-021 RSH0086         R. Scott Hanna          23-Nov-1983
0000     86 ;               Remove all timeout checks in SIP_LOOKUP_QFILE.
0000     87 ;
0000     88 ;       V03-020 RSH0080         R. Scott Hanna          11-Nov-1983
0000     89 ;               Use SIP_A_INDEXFHDR and SIP_A_FILEHDR as the index file
0000     90 ;               header and file header buffers in the call to FIL$OPENFILE_1
0000     91 ;               from SIP_LOOKUP_QFILE.
0000     92 ;
0000     93 ;       V03-019 TMK0001         Todd M. Katz            08-Nov-1983
0000     94 ;               Add a PQL$_JTQUOTA (job-wide logical name table creation
0000     95 ;               quota) quota item to the standalone configure process's
0000     96 ;               $CREPRC quota list.
0000     97 ;
0000     98 ;       V03-018 WMC0006         Wayne Cardoza           13-Oct-1983
0000     99 ;               Better error reporting on file open errors.
0000    100 ;
0000    101 ;       V03-017 DWT0126         David W. Thiel          12-Sep-1983
0000    102 ;               Define system time early without writing anything to
0000    103 ;               the system disk.  Set cluster-wide time when joining or
0000    104 ;               forming a cluster.
0000    105 ;               Use correct synchronization when deallocating the file
0000    106 ;               cache.
0000    107 ;               Remove temporary crock to force use of XQP with system
0000    108 ;               disk.
0000    109 ;
0000    110 ;       V03-016 RSH0058         R. Scott Hanna          24-Aug-1983
0000    111 ;               Add the routine SIP_LOOKUP_QFILE. This routine attempts
0000    112 ;               to open the disk quorum file using FILEREAD.
0000    113 ;
0000    114 ;       V03-015 TCM0001         Trudy C. Matthews       08-Aug-1983
```

```
0000    115 ;
0000    116 ;               Take out a shared lock on the system disk as soon as locking
0000    117 ;               is enabled.
0000    118 ;
0000    119 ;       V03-014 WMC0005         Wayne Cardoza           06-Aug-1983
0000    120 ;               Logical names not available when STACONFIG started.
0000    121 ;               STACONFIG needs all privileges.
0000    122 ;
0000    123 ;       V03-013 WMC0004         Wayne Cardoza           01-Aug-1983
0000    124 ;               Boot with an XQP system disk.
0000    125 ;
0000    126 ;       V03-012 DWT0112         David W. Thiel          29-July-1983
0000    127 ;               Add stand-alone configure invocation, lock state
0000    128 ;               setting, and waiting for cluster formation.
0000    129 ;
0000    130 ;       V03-011 ACG0344         Andrew C. Goldstein,    21-Jul-1983  16:40
0000    131 ;               Do mount of system disk in exec mode
0000    132 ;
0000    133 ;       V03-010 KDM0057         Kathleen D. Morse       12-Jul-1983
0000    134 ;               Change SIP_SETTIME into a loadable, cpu-dependent
0000    135 ;               routine, EXE$INIT_TODR.
0000    136 ;
0000    137 ;       V03-009 LJK0222         Lawrence J. Kenah       5-Jul-1983
0000    138 ;               Correct bug in $ENQW call introduced in LJK0211.
0000    139 ;
0000    140 ;       V03-008 LJK0211         Lawrence J. Kenah       22-Jun-1983
0000    141 ;               Several changes related to the new image activator and INSTALL
0000    142 ;
0000    143 ;               Remove the code that handcrafts a known file entry for the
0000    144 ;               ACP image. The process based XQP makes this unnecessary.
0000    145 ;
0000    146 ;               Remove the code that initializes the various KFE lists. This
0000    147 ;               is now done by INSTALL.
0000    148 ;
0000    149 ;               Add code to take out a lock for the system ID resource.
0000    150 ;
0000    151 ;               Change the name of a routine in the exec to FIL$INIWCB.
0000    152 ;
0000    153 ;       V03-007 WMC0003         Wayne Cardoza           10-May-1983
0000    154 ;               Use EXE$SYS_SECTION to map system sections.
0000    155 ;
0000    156 ;       V03-006 WMC0002         Wayne Cardoza           09-May-1983
0000    157 ;               Map the XQP image sections.
0000    158 ;
0000    159 ;       V03-005 JWH0204         Jeffrey W. Horn         28-Mar-1983
0000    160 ;               Replace BOO$CRMPSC with EXE$LOAD_PAGED.
0000    161 ;
0000    162 ;       V03-004 WMC0001         Wayne Cardoza           08-Mar-1983
0000    163 ;               Save the system boot time.
0000    164 ;               If no TOY clock, increment time by 10 msec
0000    165 ;
0000    166 ;       V03-003 ACG53600        Andrew C. Goldstein,    10-Feb-1983  17:08
0000    167 ;               Make time validation checks more liberal
0000    168 ;
0000    169 ;--
```

```
0000    172          .SBTTL  DECLARATIONS
0000    173          .nocross
0000    174
0000    175 ;
0000    176 ; MACROS:
0000    177 ;
0000    178 ;
0000    179 ; PROGRAM SECTION DEFINITION MACROS
0000    180 ;       ARGUMENTS ARE:
0000    181 ;               1) SECTION NAME (KEY WORD IS NAME)
0000    182 ;               2) ALIGNMENT    (KEY WORD IS ALIGN)
0000    183 ;
0000    184 ;       IN ALL CASE, ARGUMENTS ARE OPTIONAL
0000    185 ;
0000    186 ; MACRO TO GENERATE A PROGRAM SECTION FOR EXECUTABLE CODE
0000    187 ;
0000    188          .MACRO  PURE_SECT NAME=SIP_PURE,ALIGN=BYTE
0000    189
0000    190          .PSECT  NAME    EXE,RD,NOWRT,ALIGN
0000    191
0000    192          .ENDM   PURE_SECT
0000    193 ;
0000    194 ; MACRO TO GENERATE IMPURE DATA SEGMENT
0000    195 ;
0000    196          .MACRO  IMPURE_DATA  NAME=SIP_RWDATA,ALIGN=LONG
0000    197
0000    198          .PSECT  NAME    NOEXE,WRT,RD,ALIGN
0000    199
0000    200          .ENDM   IMPURE_DATA
0000    201 ;
0000    202 ; MACRO TO GENERATE A STRING WITH DESCRIPTOR
0000    203 ;
0000    204 ;       STRING_DESC <STRING>
0000    205 ;
0000    206 ; WHERE:
0000    207 ;       <STRING> IS THE STRING TO BE USED
0000    208 ;
0000    209          .MACRO  STRING_DESC ST,?L1,?L2
0000    210
0000    211          .LONG   L2-L1
0000    212          .LONG   L1
0000    213 L1:      .ASCII  \ST\
0000    214 L2:
0000    215
0000    216          .ENDM
0000    217
0000    218 ;
0000    219 ; MACRO TO GENERATE A LIST OF SELFRELATIVE WORD POINTERS
0000    220 ;
0000    221          .MACRO  OFFSET  LIST
0000    222          .IRP    $$$,<LIST>
0000    223          .WORD   <$$$-.-2>
0000    224          .ENDR
0000    225          .ENDM   OFFSET
0000    226
0000    227 ;
0000    228 ; EQUATED SYMBOLS:
```

```
                                M 15
0000    229 ;
0000    230         $ATRDEF                         ; FILE ATTRIBUTE DEFINITIONS
0000    231         $BOODEF                         ; BOOT CONTROL BLOCK DEFINITIONS
0000    232         $CCBDEF                         ; CHANNEL CONTROL BLOCK DEFINITIONS
0000    233         $CLUBDEF                        ; CLUSTER BLOCK DEFINITION
0000    234         $CLUDCBDEF                      ; CLUSTER QUORUM DISK CONTROL BLOCK DEFINITI
0000    235         $DEVDEF                         ; DEVICE BIT DEFINITIONS
0000    236         $DMPDEF                         ; SYSTEM DUMP FILE HEADER DEFINITONS
0000    237         $DVIDEF                         ; $GETDVI ITEM LIST CODES
0000    238         $DYNDEF                         ; STRUCTURE TYPE DEFINITIONS
0000    239         $EMBDEF CR                      ; ERROR LOG MESSAGE BUFFER FORMAT
0000    240         $ERLDEF                         ; ERROR LOG BUFFER DEFINITIONS
0000    241         $FIDDEF                         ; FILE ID OFFSET DEFINITIONS
0000    242         $IACDEF                         ; IMAGE ACTIVATOR INTERFACE BITS
0000    243         $IHDDEF                         ; IMAGE FILE HEADER DEFINITIONS
0000    244         $IHPDEF                         ; IMAGE HEADER PATCH DEFINITIONS
0000    245         $IHSDEF                         ; IMAGE HEADER SYMBOLIC DEBUGGING DEFS
0000    246         $IODEF                          ; DEFINE I/O FUNCTION CODES
0000    247         $IPLDEF                         ; DEFINE INTERRUPT PRIORITY LEVELS
0000    248         $ISDDEF                         ; IMAGE SECTION DESCRIPTIONS
0000    249         $LCKDEF                         ; FLAG BITS FOR CALL TO $ENQW
0000    250         $LNMDEF                         ; DEFINE LOG OFFSETS
0000    251         $PCBDEF                         ; DEFINE PCB OFFSETS
0000    252         $PFLDEF                         ; PAGE FILE OFFSET DEFINITONS
0000    253         $PHDDEF                         ; DEFINE PROCESS HEADER OFFSETS
0000    254         $PQLDEF                         ; PROCESS QUOTA DEFINITIONS
0000    255         $PRDEF                          ; PROCESSOR REGISTER DEFINITIONS
0000    256         $PRTDEF                         ; PAGE PROTECTION DEFINITIONS
0000    257         $PRVDEF                         ; PRIVILEGE DEFINITIONS
0000    258         $PSLDEF                         ; PSL DFINITIONS
0000    259         $PTEDEF                         ; PAGE TABLE ENTRY DEFINITIONS
0000    260         $PTRDEF                         ; POINTER CONTROL BLOCK OFFSETS
0000    261         $RPBDEF                         ; DEFINE RPB OFFSETS
0000    262         $SECDEF                         ; DEFINE PROCESS SECTION
0000    263         $TQEDEF                         ; DEFINE TIMER QUEUE ENTRY OFFSETS
0000    264         $UCBDEF                         ; UNIT CONTROL BLOCK DEFINITIONS
0000    265         $VADEF                          ; DEFINE VIRTUAL ADDRESS FIELDS
0000    266         $WCBDEF                         ; WINDOW CONTROL BLOCK DEFINITIONS
0000    267
00000002 0000   268         SIP_C_DUMPVER = 2               ; DUMP FILE HEADER VERSION
000001C4 0000   269         SIP_C_MINPAGFIL = 2500-2048     ; MINIMUM PAGE FILE REQUIRED
0000    270 ;
0000    271 ; OFFSETS INTO FILE ATTRIBUTES ARRAY
0000    272 ;
0000    273         $OFFSET 0,POSITIVE,<-
0000    274         <STATBLK,0>,-                   ; 8 BYTE STATISTICS BLOCK CONSISTING OF
0000    275         FILELBN,-                       ; STARTING LBN OR 0 IF NOT CONTIG
0000    276         FILESIZE,-                      ; SIZE OF FILE IN 512 BYTE BLOCKS
0000    277         IMAGEVBN,-                      ; FIRST VBN IF IMAGE FORMAT
0000    278         IMAGESIZE,-                     ; SIZE IF IMAGE FORMAT
0000    279         RTRVLEN,-                       ; BYTE COUNT OF RETRIEVAL POINTERS
0000    280         <RTRVPTRS,0>-                   ; FIRST RETRIEVAL POINTER
0000    281         >
0000        STATBLK:
0000        FILELBN:
0004        FILESIZE:
0008        IMAGEVBN:
```

```
                        000C        IMAGESIZE:
                        0010        RTRVLEN:
                        0014        RTRVPTRS:
                        0000    282
                        0000    283         .WEAK   XDT$START               ; IF DEBUGGING, THEN DEFINED
                        0000    284
                        0000    285         .cross
                        0000    286
                        0000    287 ;
                        0000    288 ; OWN STORAGE:
                        0000    289 ;
                        0000    290         PURE_SECT
                        0000    291
                        0000    292 SIP_Q_TTNAME:
                        0000    293         STRING_DESC     <OPA0>          ; DEVICE NAME FOR TERMINAL
                        000C    294
                        000C    295 SIP_Q_FIBDESC:
000000CC'00000010'      000C    296         .LONG   SIP_C_FIB_SIZE,SIP_A_FIB ; DESCRIPTOR FOR FILE IDENT BLOCK
                        0014    297 SIP_A_ATRLIST:
              0010 0056 0014    298         .WORD   ATR$S_ASCNAME,ATR$C_ASCNAME ; ASCII NAME ATTRIBUTE
          00000000'     0018    299         .LONG   SIP_A_ERLBUFFER         ; SET ADR TO STORE NAME HERE
          00000000      001C    300         .LONG   0                       ; END OF ATTRIBUTE LIST
                        0020    301
                        0020    302 SIP_Q_STARTUP:                          ; STARTUP PROCESS NAME
                        0020    303         STRING_DESC     <STARTUP>       ;
                        002F    304
                        002F    305
                        002F    306 SIP_Q_SPOUTPUT:                         ; STARTUP PROCESS OUTPUT
                        002F    307         STRING_DESC     <OPA0:>         ; CONSOLE
                        003C    308 SIP_Q_SPOUTXDT:                         ; STARTUP PROCESS OUTPUT (DELTA)
                        003C    309         STRING_DESC     <NLA0:>         ; NULL DEVICE
                        0049    310
                        0049    311 SIP_Q_SPIMAGE:                          ; STARTUP PROCESS IMAGE
                        0049    312         STRING_DESC     <SYS$SYSTEM:LOGINOUT.EXE>       ; NORMAL LOGINOUT IMAGE
                        0068    313
                        0068    314 SIP_Q_PRVMSK:
            FFFFFFFF FFFFFFFF   0068    315         .LONG   -1,-1                   ; INITIAL PRIVILEGES
                        0070    316
                        0070    317 FAOERR: STRING_DESC  <%SYSINIT-E- !AC, status = !XL>
                        0095    318 CRELNMERR:
63 20 6F 74 20 64 65 6C 69 61 66 00'    0095    319         .ASCIC  \failed to create system logical names\
6D 65 74 73 79 73 20 65 74 61 65 72     00A1
6D 61 6E 20 6C 61 63 69 67 6F 6C 20     00AD
                              73 65     00B9
                                 25     0095
                        00BB    320 PAGFILERR:
6C 69 61 66 20 70 75 6B 6F 6F 6C 00'    00BB    321         .ASCIC  \lookup failure on paging file\
6E 69 67 61 70 20 6E 6F 20 65 72 75     00C7
                  65 6C 69 66 20 67     00D3
                                 1D     00BB
                        00D9    322 MSGFILERR:
6C 69 66 20 65 67 61 73 73 65 6D 00'    00D9    323         .ASCIC  \message file not found, or insufficient SPT to map it\
2C 64 6E 75 6F 66 20 74 6F 6E 20 65     00E5
63 69 66 66 75 73 6E 69 20 72 6F 20     00F1
20 6F 74 20 54 50 53 20 74 6E 65 69     00FD
                  74 69 20 70 61 6D     0109
                                 35     00D9
```

```
                                                010F   324
                                                010F   325 ACPINIERR:
74 69 6E 69 20 50 43 41 31 31 46 00'            010F   326         .ASCIC  \F11ACP initialization error\
65 20 6E 6F 69 74 61 7A 69 6C 61 69             011B
                        72 6F 72 72             0127
                                    1B          010F
                                                012B   327
74 6E 75 6F 6D 20 72 6F 72 72 65 00'            012B   328 MOUERR: .ASCIC  \error mounting system device\
64 20 6D 65 74 73 79 73 20 67 6E 69             0137
                        65 63 69 76 65          0143
                                    1C          012B
                                                0148   329
                                                0148   330 LOCKERR:
6E 69 6B 61 74 20 72 6F 72 72 65 00'            0148   331         .ASCIC  \error taking out lock on system disk\
6F 20 6B 63 6F 6C 20 74 75 6F 20 67             0154
73 69 64 20 6D 65 74 73 79 73 20 6E             0160
                                    6B          016C
                                    24          0148
                                                016D   332
                                                016D   333 INIPAGFIL:                       ; ERROR INITIALIZING THE PAGE OR SWAP FILE
6F 20 65 6C 69 66 20 65 67 61 70 00'            016D   334         .ASCIC  \page file or swap file control block initialization error\
20 65 6C 69 66 20 70 61 77 73 20 72             0179
63 6F 6C 62 20 6C 6F 72 74 6E 6F 63             0185
61 7A 69 6C 61 69 74 69 6E 69 20 6B             0191
        72 6F 72 72 65 20 6E 6F 69 74           019D
                                    39          016D
                                                01A7   335
                                                01A7   336 RMSMAPERR:                       ; ERROR ON RMS FILE MAP
74 6F 6E 20 45 58 45 2E 53 4D 52 00'            01A7   337         .ASCIC  \RMS.EXE not found, or insufficient SPT to map it\
69 20 72 6F 20 2C 64 6E 75 6F 66 20             01B3
20 74 6E 65 69 63 69 66 66 75 73 6E             01BF
69 20 70 61 6D 20 6F 74 20 54 50 53             01CB
                                    74          01D7
                                    30          01A7
                                                01D8   338
                                                01D8   339 FILOPNERR:                       ; ANY FILE OPEN ERROR - MORE MESSAGES LATER
69 6E 65 70 6F 20 72 6F 72 72 65 00'            01D8   340         .ASCIC  /error opening file/
                        65 6C 69 66 20 67 6E    01E4
                                    12          01D8
                                                01EB   341
                                                01EB   342 INIWCBERR:                       ; ERROR INITING A WINDOW CONTROL BLOCK
69 74 69 6E 69 20 72 6F 72 72 65 00'            01EB   343         .ASCIC  \error initializing a window control block\
69 77 20 61 20 67 6E 69 7A 69 6C 61             01F7
6C 6F 72 74 6E 6F 63 20 77 6F 64 6E             0203
                        6B 63 6F 6C 62 20       020F
                                    29          01EB
                                                0215   344
69 6E 65 70 6F 20 72 6F 72 72 65 00'            0215   345 XQPERR: .ASCIC  /error opening or mapping F11BXQP/
6E 69 70 70 61 6D 20 72 6F 20 67 6E             0221
          50 51 58 42 31 31 46 20 67            022D
                                    20          0215
                                                0236   346
                                                0236   347 SYSID_LOCK_ERR:
6F 20 6F 74 20 65 6C 62 61 6E 75 00'            0236   348         .ASCIC  \unable to obtain lock for system ID resource\
66 20 6B 63 6F 6C 20 6E 69 61 74 62             0242
44 49 20 6D 65 74 73 79 73 20 72 6F             024E
          65 63 72 75 6F 73 65 72 20            025A
```

```
                                    2C  0236
                                        0263    349
                                        0263    350 SIP_CLU_MSG:
20 6F 74 20 67 6E 69 74 69 61 77 00'    0263    351         .ASCIC  \waiting to form or join VAXcluster\
6E 69 6F 6A 20 72 6F 20 6D 72 6F 66     026F
   72 65 74 73 75 6C 63 58 41 56 20     027B
                                    22  0263
                                        0286    352
                                        0286    353 INIKNOWNFIL:
20 65 6C 69 66 20 6E 77 6F 6E 6B 00'    0286    354         .ASCIC  \known file list initialization error\
6C 61 69 74 69 6E 69 20 74 73 69 6C     0292
6F 72 72 65 20 6E 6F 69 74 61 7A 69     029E
                                    72  02AA
                                    24  0286
                                        02AB    355
                                        02AB    356 PAGFILNAM:
59 53 2E 45 4C 49 46 45 47 41 50 00'    02AB    357         .ASCIC  \PAGEFILE.SYS\
                                    53  02B7
                                    0C  02AB
                                        02B8    358 SWPFILNAM:
59 53 2E 45 4C 49 46 50 41 57 53 00'    02B8    359         .ASCIC  \SWAPFILE.SYS\
                                    53  02C4
                                    0C  02B8
                                        02C5    360 RMSFILNAM:
            45 58 45 2E 53 4D 52 00'    02C5    361         .ASCIC  \RMS.EXE\
                                    07  02C5
                                        02CD    362 MSGFILNAM:
3A 45 47 41 53 53 45 4D 24 53 59 53     02CD    363         .ASCII  \SYS$MESSAGE:SYSMSG.EXE\
   45 58 45 2E 47 53 4D 53 59 53        02D9
                        00000016    02E3    364         MSGFILNAMSZ=.-MSGFILNAM
                                    02E3    365
                                    02E3    366 ;
                                    02E3    367 ; ***** PAGE FILE MUST BE FIRST
                                    02E3    368 ;
                                    02E3    369 SIP_A_NAMES:
                        000002AB'   02E3    370         .LONG   PAGFILNAM               ; FILENAME AND ERROR POINTER
                        000002B8'   02E7    371         .LONG   SWPFILNAM               ;
                        000002C5'   02EB    372         .LONG   RMSFILNAM               ;
                        00000000    02EF    373         .LONG   0                       ; END OF LIST
                                    02F3    374 ;
46 3A 4D 45 54 53 59 53 24 53 59 53 02F3    375 XQPNAM: .ASCII  /SYS$SYSTEM:F11BXQP.EXE/
   45 58 45 2E 50 51 58 42 31 31    02FF
                        00000016    0309    376         XQPNAMSIZ = .-XQPNAM
                                    0309    377 ;
                                    0309    378 ;
                                    0309    379         IMPURE_DATA SIP_RWDATA_PAGE,PAGE
                                    0000    380 ;
                                    0000    381 ;       THIS BUFFER IS USED FOR THE QUORUM FILE LOOKUP AND TO READ
                                    0000    382 ;       THE SYSTEM DUMP FILE FOR ERROR LOG INFORMATION
                                    0000    383 ;
                                    0000    384 SIP_A_ERLBUFFER:                    ;
                        00000600    0000    385         .BLKB   <3*512>             ; 3 PAGES
                        00000000    0600    386 SIP_A_INDEXFHDR = SIP_A_ERLBUFFER   ; INDEX FILE HEADER BUF (FIL$OPENFILE)
                        00000200    0600    387 SIP_A_FILEHDR   = SIP_A_ERLBUFFER+512   ; FILE HEADER BUFFER (FIL$OPENFILE)
                                    0600    388
                                    0600    389         IMPURE_DATA
                                    0000    390
```

```
                     0000   391 MSGFILFAB:      $FAB    FAC=GET,-           ; FILE ACCESS IS GET (READ)
                     0000   392                 FOP=<UFO>,-                 ; USER FILE OPEN
                     0000   393                 FNA=MSGFILNAM,-             ; ADDRESS OF FILE NAME STRING
                     0000   394                 FNS=MSGFILNAMSZ,-           ;
                     0000   395                 RFM=FIX,-                   ; FIXED RECORD FORMAT
                     0000   396                 MRS=512,-                   ; MAXIMUM RECORD SIZE OF ONE PAGE
                     0000   397                 RTV=255,-                   ; LET ACP COMPUTE LARGEST RETRIEVAL WINDOW
                     0000   398                 XAB=MSGFILXAB               ; EXTENDED ATTRIBUTE BLOCK
                     0050   399 MSGFILXAB:      $XABFHC                     ; EXTENDED ATTRIBUTE BLOCK FOR FILE HEADER
                     007C   400
                     007C   401 XQPFAB: $FAB    FAC=GET,-                   ; FILE ACCESS IS GET (READ)
                     007C   402                 FOP=<UFO>,-                 ; USER FILE OPEN
                     007C   403                 FNA=XQPNAM,-                ; ADDRESS OF FILE NAME STRING
                     007C   404                 FNS=XQPNAMSIZ,-             ;
                     007C   405                 RFM=FIX,-                   ; FIXED RECORD FORMAT
                     007C   406                 MRS=512,-                   ; MAXIMUM RECORD SIZE OF ONE PAGE
                     007C   407                 RTV=255                     ; LET ACP COMPUTE LARGEST RETRIEVAL WINDOW
                     00CC   408
                     00CC   409 SIP_A_FIB:                                  ; FILE IDENTIFICATION BLOCK
           00000000  00CC   410         .LONG   0                           ; ACCESS CONTROL INFORMATION
       0000 0000 0000  00D0   411         .WORD   0,0,0                       ; RETURNED FILE ID
       0000 0004 0004  00D6   412         .WORD   FID$C_MFD,FID$C_MFD,0       ; DIRECTORY ID OF MFD
           00000010  00DC   413         SIP_C_FIB_SIZE=.-SIP_A_FIB
                     00DC   414 SIP_L_TTCHAN:
           000000E0  00DC   415         .BLKL   1                           ; CHANNEL FOR TERMINAL HERE
                     00E0   416
                     00E0   417 SIP_Q_RETADR:                               ; RETURN ADDRESS RANGE FROM EXPREG
           000000E8  00E0   418         .BLKQ   1
                     00E8   419 SIP_Q_TMPDESC:                              ; TEMPORY STRING DESCRIPTOR
           000000F0  00E8   420         .BLKQ   1
                     00F0   421 SIP_Q_STATBLK:
           000000F8  00F0   422         .BLKQ   1                           ; STATISTICS BLOCK RETURNED BY FIL$OPENFILE
                     00F8   423 SIP_Q_RTRVBUF:                              ; DESCRIPTOR FOR RTRV PTR BUFFER
           00000100  00F8   424         .BLKQ   1
                     0100   425 SIP_L_RTRVLEN:                              ; RETURNED RTRV PTR BUFFER LENGTH
           00000104  0100   426         .BLKL   1
                     0104   427 SIP_A_OPENARG:                              ; ARGUMENT LIST TO FIL$OPENFILE
           00000007  0104   428         .LONG   7                           ; 7 ARGUMENTS TO FIL$OPENFILE
           00000134' 0108   429         .LONG   SIP_L_DSKCHAN               ; ADDRESS TO RETURN DISK CHANNEL
           000000E8' 010C   430         .LONG   SIP_Q_TMPDESC               ; ADDRESS OF FILE NAME DESCRIPTOR
           00000000' 0110   431         .LONG   SIP_A_INDEXFHDR             ; BUFFER ADDRESS FOR INDEX FILE HEADER
           00000200' 0114   432         .LONG   SIP_A_FILEHDR               ; BUFFER ADDRESS FOR FILE HEADER
           000000F0' 0118   433         .LONG   SIP_Q_STATBLK               ; ADDRESS TO RETURN STATISTICS BLOCK
                     011C   434                                             ;    STARTING LBN IF CONTIG, 0 IF NOT
                     011C   435                                             ;    FILE SIZE IN BLOCKS
           00000100' 011C   436         .LONG   SIP_L_RTRVLEN               ; ADR TO RETURN RTRV PTR BUF LENGTH
           000000F8' 0120   437         .LONG   SIP_Q_RTRVBUF               ; ADR OF RTRV PTR BUF DESCRIPTOR
                     0124   438
                     0124   439 SIP_L_ERRSEQ:
           00000000  0124   440         .LONG   0                           ; SAVED ERROR SEQUENCE NUMBER
                     0128   441                                             ; FROM DUMP FILE HEADER
                     0128   442 SIP_A_FILATT:                               ; LIST OF FILE ATTRIBUTE AREAS
                     0128   443 SIP_L_PAGATT:                               ; PAGE FILE
           0000012C  0128   444         .BLKL   1
                     012C   445 SIP_L_SWPATT:                               ; SWAP FILE
           00000130  012C   446         .BLKL   1
                     0130   447 SIP_L_RMSATT:                               ; RMS
```

```
                        00000134  0130    448          .BLKL   1
                                  0134    449
                                  0134    450 SIP_L_DSKCHAN:
                        00000138  0134    451          .BLKL   1                       ; CHANNEL FOR DISK HERE
                                  0138    452
                                  0138    453 SIP_Q_LINBUF:
                      0084'0000   0138    454          .WORD   0,SIP_C_LINBUFSIZ       ; DESCRIPTOR FOR LINE BUFFER
                       00000140'  013C    455          .LONG   SIP_T_LINBUF
                                  0140    456
                                  0140    457 SIP_T_LINBUF:
                        000001C4  0140    458          .BLKB   132
                                  01C4    459
                        00000084  01C4    460 SIP_C_LINBUFSIZ=.-SIP_T_LINBUF
                                  01C4    461
                                  01C4    462 CREPRCERR:                              ; CREATE PROCESS ERROR
                             27'  01C4    463          .BYTE   CREERREND-.-1          ; LENGTH OF STRING
65 63 6F 72 70 20 65 74 61 65 72 63 01C5    464          .ASCII  \create process error on \
20 6E 6F 20 72 6F 72 72 65 20 73 73 01D1
                                  01DD    465 CREPRCNAM:                              ;
                        000001EC  01DD    466          .BLKB   15                     ;
                                  01EC    467 CREERREND:
                                  01EC    468
                                  01EC    469 SIP_Q_SPINPUT:                          ; STARTUP PROCESS INPUT
                        00000000  01EC    470          .LONG   0                      ; COUNT FOR STRING
                       00000001'  01F0    471          .LONG   EXE$GT_STARTUP+1       ; ADDRESS
                                  01F4    472
                                  01F4    473 XQP_GSDNAM:
      30 30 30 5F 50 51 58 53 59 53 01F4    474          .ASCII  /SYSXQP_000/
                        0000000A  01FE    475          XQP_GSDNAM_SIZ = .-XQP_GSDNAM
                                  01FE    476 XQP_GSD_DESC:
                        0000000A  01FE    477          .LONG   XQP_GSDNAM_SIZ
                       000001F4'  0202    478          .ADDRESS XQP_GSDNAM
                                  0206    479 XQP_NAME:
59 53 24 53 59 53 0000020E'010E0000' 0206 480          .ASCID /SYS$SYSTEM:F11BXQP.EXE/
50 51 58 42 31 31 46 3A 4D 45 54 53 0214
                     45 58 45 2E  0220
                                  0224    481 XQP_DEF:
59 53 24 53 59 53 0000022C'010E0000' 0224 482          .ASCID /SYS$SYSTEM:.EXE/
         45 58 45 2E 3A 4D 45 54 53 0232
                                  023B    483 XQP_INADDR:
                  00000000 00000000 023B    484          .LONG 0,0
                                  0243    485 XQP_RETADDR:
                  00000000 00000000 0243    486          .LONG 0,0
                                  024B    487 XQP_HEADER:
                        0000044B  024B    488          .BLKB 512
```

```
                        044B    490          .SUBTITLE       Data Used by $ENQW Request
                        044B    491 ;+
                        044B    492 ; The following data area is used by the $ENQW request that obtains a lock
                        044B    493 ; whose name contains the system ID
                        044B    494 ;-
                        044B    495
                        044B    496 LOCK_FLAGS = -                              ; Flags used by $ENQW call
                        044B    497             LCK$M_SYSTEM ! -                ;  Do not qualify lock name with UIC
                        044B    498             LCK$M_NOQUEUE ! -               ;  There should be nothing to wait for
                        044B    499             LCK$M_CVTSYS ! -                ;  The lock will be owned by the system
                 0000005C   044B    500             LCK$M_SYNCSTS
                        044B    501
                        044B    502 ; Lock status block. The lock ID will be stored in an exec data cell after
                        044B    503 ; the service successfully completes.
                        044B    504
                        044B    505 LOCK_STATUS_BLOCK:
                 0000044F   044B    506 LOCK_STATUS:    .BLKW   2
                 00000000   044F    507 LOCK_ID:        .LONG   0
                        0453    508
                        0453    509 ; The lock name begins with the facility name in ASCII. The guts of the lock
                        0453    510 ; name consists of the six-byte system ID. The "ID" suffix is a cute way of
                        0453    511 ; rounding the name up to multiple of four.
                        0453    512
 44 49 5F 53 59 53 24 53 59 53  0453    513 LOCK_NAME:      .ASCII  /SYS$SYS_ID/
                 00000463   045D    514 SYS_ID:         .BLKB   6
                 00000010   0463    515 LOCK_NAME_SIZE = . - LOCK_NAME
                        0463    516
                        0463    517 LOCK_NAME_DESC:
                 00000010   0463    518             .LONG           LOCK_NAME_SIZE
                 00000453'  0467    519             .ADDRESS        LOCK_NAME
```

```
                                046B    521          .SUBTITLE       Data Used To Create Stand-Alone Configure Process
                                046B    522
                                046B    523 ;
                                046B    524 ; The following data is used in creating the stand-alone Configure Process
                                046B    525 ;
                                046B    526
                                046B    527 ; Image name
                                046B    528 ;
4E 4F 43 41 54 53 00000473'010E0000'046B    529 STAC_IMAGE:     .ASCID  /STACONFIG.EXE/
                45 58 45 2E 47 49 46  0479
                                0480    530
                                0480    531 ; Input/output/error names
                                0480    532 ;
3A 30 41 50 4F 5F 00000488'010E0000'0480    533 STAC_OPER:      .ASCID  /_OPA0:/
                                048E    534
                                048E    535 ; Process privilege mask
                                048E    536 ;
                    FFFFFFFF FFFFFFFF   048E    537 STAC_PRV_MSK:   .LONG   -1,-1
                                0496    538
                                0496    539 ; Process name
                                0496    540 ;
4E 4F 43 41 54 53 0000049E'010E0000'0496    541 STAC_PRC:       .ASCID  /STACONFIG/
                            47 49 46  04A4
                                04A7    542
                                04A7    543 ; Process quotas
                                04A7    544 ;
                                04A7    545 STAC_QLIST:
                             01 04A7    546          .BYTE   PQL$_ASTLM
                       000000C8 04A8    547          .LONG   200
                             02 04AC    548          .BYTE   PQL$_BIOLM
                       000000C8 04AD    549          .LONG   200
                             03 04B1    550          .BYTE   PQL$_BYTLM
                       000186A0 04B2    551          .LONG   100000
                             04 04B6    552          .BYTE   PQL$_CPULM
                       00000000 04B7    553          .LONG   0
                             05 04BB    554          .BYTE   PQL$_DIOLM
                       000000C8 04BC    555          .LONG   200
                             0C 04C0    556          .BYTE   PQL$_ENQLM
                       000000C8 04C1    557          .LONG   200
                             06 04C5    558          .BYTE   PQL$_FILLM
                       000000C8 04C6    559          .LONG   200
                             07 04CA    560          .BYTE   PQL$_PGFLQUOTA
                       00005000 04CB    561          .LONG   20480
                             08 04CF    562          .BYTE   PQL$_PRCLM
                       000000C8 04D0    563          .LONG   200
                             09 04D4    564          .BYTE   PQL$_TQELM
                       000000C8 04D5    565          .LONG   200
                             0B 04D9    566          .BYTE   PQL$_WSDEFAULT
                       00000064 04DA    567          .LONG   100
                             0A 04DE    568          .BYTE   PQL$_WSQUOTA
                       00000200 04DF    569          .LONG   512
                             0E 04E3    570          .BYTE   PQL$_JTQUOTA
                       00000400 04E4    571          .LONG   1024
                             00 04E8    572          .BYTE   PQL$_LISTEND
                                04E9    573
                                04E9    574 SIP_CLU_TIMOUT:                     ; 100 milli-second quadword value
                    FFFFFFFF FFF0BDC0   04E9    575          .LONG   -1000*1000,-1
```

```
                                04F1    577          .SUBTITLE       Data Used For Quorum disk
                                04F1    578
                                04F1    579 SIP_QD_CHAN:                                ; Quorum disk channel number
                    00000000    04F1    580          .LONG   0
                                04F5    581
                                04F5    582 SIP_QD_IOSB:                                ; I/O status block
                                04F5    583 SIP_QD_STATBUF:                             ; Statistics buffer
                00000000 00000000   04F5    584          .QUAD   0
                                04FD    585
                                04FD    586 SIP_QD_DESCR:                               ; Quorum disk name descriptor
                        0010    04FD    587          .WORD   CLUDCB$S_DISK_QUORUM
                          00'   04FF    588          .BYTE   DSC$K_DTYPE_T
                          00'   0500    589          .BYTE   DSC$K_CLASS_S
                    00000000'   0501    590          .LONG   CLU$GB_QDISK
                                0505    591
                                0505    592 SIP_QF_DESCR:                               ; Full quorum file name descriptor
                        0000    0505    593          .WORD   0
                          00'   0507    594          .BYTE   DSC$K_DTYPE_T
                          00'   0508    595          .BYTE   DSC$K_CLASS_S
                    00000521'   0509    596          .LONG   SIP_QF_BUFFER
                                050D    597
                                050D    598 SIP_QF_NAME:
52 4F 55 51 5D 30 30 30 30 30 30 5B 050D    599          .ASCII  /[000000]QUORUM.DAT;1/
            31 3B 54 41 44 2E 4D 55 0519
                    00000014    0521    600          SIP_QF_NAME_SIZE = .-SIP_QF_NAME
                                0521    601
                                0521    602 SIP_QF_BUFFER:
                    00000575    0521    603          .BLKB   64+SIP_QF_NAME_SIZE
                                0575    604
                                0575    605 SIP_QD_ITMLST:
                00E8 0040       0575    606          .WORD   64,DVI$_FULLDEVNAM
                 00000521'      0579    607          .LONG   SIP_QF_BUFFER
                 00000505'      057D    608          .LONG   SIP_QF_DESCR
                 00000000       0581    609          .LONG   0
```

```
                        05B5   611          .SBTTL  IMPURE DATA FOR $CRELNM AND $TRNLNM CALLS
                        05B5   612
                        05B5   613 SYS_COMMON_ITMLST:
                0003 0004 05B5   614          .WORD   4,LNM$_ATTRIBUTES
                 0000042F' 05B9   615          .LONG   TERMINAL_CONCEALED_ATTR ;SYS$COMMON BOTH TERMINAL AND CONCEALED
                 00000000  05BD   616          .LONG   0
                        05C1   617 SYS_SYSROOT_CMNSYS_LEN:
                    0000  05C1   618          .WORD   0
                    0002  05C3   619          .WORD   LNM$_STRING
                        05C5   620 SYS_SYSROOT_CMNSYS:
                 00000000  05C5   621          .LONG   0
            00000000 00000000  05C9   622          .QUAD   0
                        05D1   623
                        05D1   624 SYS_SYSDEVICE_ITMLST:
                0003 0004 05D1   625          .WORD   4,LNM$_ATTRIBUTES
                 000005BD' 05D5   626          .LONG   SYS_SYSDEVICE_ATTR
                 00000000  05D9   627          .LONG   0
                        05DD   628 SYS_SYSDEVICE_DEV_LEN:
                    0020  05DD   629          .WORD   32
                    0002  05DF   630          .WORD   LNM$_STRING
                        05E1   631 SYS_SYSDEVICE_DEV:
                 00000000' 05E1   632          .LONG   SIP_A_ERLBUFFER
                 000005AD' 05E5   633          .LONG   SYS_SYSDEVICE_DEV_LEN
                 00000000  05E9   634          .LONG   0
                        05ED   635
                        05ED   636 SYS_SYSDEVICE_ATTR:
                 00000000  05ED   637          .LONG   0
                        05F1   638
                        05F1   639 SYS_SYSDEVICE_DVI_LST:
                    0020  05F1   640          .WORD   32
                    00E8  05F3   641          .WORD   DVI$_FULLDEVNAM
                 00000000' 05F5   642          .LONG   SIP_A_ERLBUFFER
                 000005AD' 05F9   643          .LONG   SYS_SYSDEVICE_DEV_LEN
                 00000000  05FD   644          .LONG   0
                        0601   645
                        0601   646 SYS_SYSROOT_ITMLST:
                0003 0004 0601   647          .WORD   4,LNM$_ATTRIBUTES
                 0000042F' 0605   648          .LONG   TERMINAL_CONCEALED_ATTR ;TOPSYS BOTH TERMINAL AND CONCEALED
                 00000000  0609   649          .LONG   0
                        060D   650 SYS_SYSROOT_TOPSYS_LEN:
                    0000  060D   651          .WORD   0
                    0002  060F   652          .WORD   LNM$_STRING
                        0611   653 SYS_SYSROOT_TOPSYS:
                 00000000  0611   654          .LONG   0
                 00000000  0615   655          .LONG   0
                0003 0004 0619   656          .WORD   4,LNM$_ATTRIBUTES
                 00000433' 061D   657          .LONG   NO_ATTR                 ;CMNSYS NEITHER TERMINAL NOR CONCEALED
                 00000000  0621   658          .LONG   0
                    000B' 0625   659          .WORD   SYS_COMMON_LENGTH
                    0002  0627   660          .WORD   LNM$_STRING
                 00000338' 0629   661          .LONG   SYS_COMMON
            00000000 00000000  062D   662          .QUAD   0
                        0635   663
                        0635   664 SYS_TOPSYS_ITMLST:
                        0635   665 SYS_TOPSYS_DIRNAM_LEN:
                    0000  0635   666          .WORD   0
                    0002  0637   667          .WORD   LNM$_STRING
```

```
                        0609   668 SYS_TOPSYS_DIRNAM:
              00000000  0609   669          .LONG   0
00000000 00000000       060D   670          .QUAD   0
```

```
                                  0615   672          .SBTTL  PURE DATA FOR $CRELNM AND $TRNLNM CALLS
                                  0615   673
                                  0615   674          PURE_SECT
                                  0309   675
                                  0309   676 CMNSYS:
5D 2E 4E 4F 4D 4D 4F 43 53 59 53 00'  0309   677          .ASCIC  /SYSCOMMON.]/
                                08    0309
                                  0315   678
                                  0315   679 LNM_FILE_DEV:
49 46 24 4D 4E 4C 0000031D'010E0000'  0315   680          .ASCID  /LNM$FILE_DEV/
                  56 45 44 5F 45 4C   0323
                                  0329   681
                                  0329   682 LNM_SYSTEM_DESC:
59 53 24 4D 4E 4C 00000331'010E0000'  0329   683          .ASCID  /LNM$SYSTEM/
                        4D 45 54 53   0337
                                  033B   684
                                  033B   685 SYS_COMMON:
   3A 4E 4F 4D 4D 4F 43 24 53 59 53   033B   686          .ASCII  /SYS$COMMON:/
                           00000008   0346   687 SYS_COMMON_LENGTH = . - SYS_COMMON
                                  0346   688
                                  0346   689 SYS_COMMON_DESC:
4F 43 24 53 59 53 0000034E'010E0000'  0346   690          .ASCID  /SYS$COMMON/
                        4E 4F 4D 4D   0354
                                  0358   691
                                  0358   692 SYS_MESSAGE:
3A 54 4F 4F 52 53 59 53 24 53 59 53   0358   693          .ASCII  /SYS$SYSROOT:[SYSMSG]/
            5D 47 53 4D 53 59 53 5B   0364
                           00000014   036C   694 SYS_MESSAGE_LEN = . - SYS_MESSAGE
                                  036C   695
                                  036C   696 SYS_MESSAGE_DESC:
45 4D 24 53 59 53 00000374'010E0000'  036C   697          .ASCID  /SYS$MESSAGE/
                        45 47 41 53 53   037A
                                  037F   698
                                  037F   699 SYS_SHARE:
3A 54 4F 4F 52 53 59 53 24 53 59 53   037F   700          .ASCII  /SYS$SYSROOT:[SYSLIB]/
            5D 42 49 4C 53 59 53 5B   038B
                           00000014   0393   701 SYS_SHARE_LEN = . - SYS_SHARE
                                  0393   702
                                  0393   703 SYS_SHARE_DESC:
48 53 24 53 59 53 0000039B'010E0000'  0393   704          .ASCID  /SYS$SHARE/
                              45 52 41   03A1
                                  03A4   705
                                  03A4   706 SYS_SYSDEVICE_DESC:
59 53 24 53 59 53 000003AC'010E0000'  03A4   707          .ASCID  /SYS$SYSDEVICE/
                        45 43 49 56 45 44 53   03B2
                                  03B9   708
                                  03B9   709 SYS_DISK_DESC:
49 44 24 53 59 53 000003C1'010E0000'  03B9   710          .ASCID  /SYS$DISK/
                                 48 53   03C7
                                  03C9   711
                                  03C9   712 SYS_SYSROOT_DESC:
59 53 24 53 59 53 000003D1'010E0000'  03C9   713          .ASCID  /SYS$SYSROOT/
                        54 4F 4F 52 53   03D7
                                  03DC   714
                                  03DC   715 SYS_SYSTEM:
3A 54 4F 4F 52 53 59 53 24 53 59 53   03DC   716          .ASCII  /SYS$SYSROOT:[SYSEXE]/
            5D 45 58 45 53 59 53 5B   03E8
```

```
                           00000014  03F0    717 SYS_SYSTEM_LEN = . - SYS_SYSTEM
                                     03F0    718
                                     03F0    719 SYS_SYSTEM_DESC:
59 53 24 53 59 53 000003F8'010E0000' 03F0    720         .ASCID  /SYS$SYSTEM/
                              4D 45 54 53  03FE
                                     0402    721
                                     0402    722 SYS_TOPSYS_DESC:
4F 54 24 53 59 53 0000040A'010E0000' 0402    723         .ASCID  /SYS$TOPSYS/
                              53 59 53 50  0410
                                     0414    724
                                     0414    725 SYSUAFALT:
         54 4C 41 46 41 55 53 59 53  0414    726         .ASCII  /SYSUAFALT/
                           00000009  041D    727 SYSUAFALT_LEN = . - SYSUAFALT
                                     041D    728
                                     041D    729 SYSUAF_DESC:
46 41 55 53 59 53 00000425'010E0000' 041D    730         .ASCID  /SYSUAF/
                                     042B    731
                           00000001  042B    732 EXEC_MODE:      .LONG   PSL$C_EXEC
                                     042F    733
                                     042F    734 TERMINAL_CONCEALED_ATTR:
                           00000300  042F    735                 .LONG   LNM$M_TERMINAL!LNM$M_CONCEALED
                                     0433    736
                           00000000  0433    737 NO_ATTR:        .LONG   0
                                     0437    738
                                     0437    739 SYS_MESSAGE_ITMLST:
                         0002 0014   0437    740         .WORD   SYS_MESSAGE_LEN,LNM$_STRING
                           00000358' 043B    741         .LONG   SYS_MESSAGE
                 00000000 00000000   043F    742         .QUAD   0
                                     0447    743
                                     0447    744 SYS_SHARE_ITMLST:
                 0002 0014           0447    745         .WORD   SYS_SHARE_LEN,LNM$_STRING
                   0000037F'         044B    746         .LONG   SYS_SHARE
        00000000 00000000            044F    747         .QUAD   0
                                     0457    748
                                     0457    749 SYS_SYSTEM_ITMLST:
                 0002 0014           0457    750         .WORD   SYS_SYSTEM_LEN,LNM$_STRING
                   000003DC'         045B    751         .LONG   SYS_SYSTEM
        00000000 00000000            045F    752         .QUAD   0
                                     0467    753
                                     0467    754 SYSUAF_ITMLST:
                 0002 0009           0467    755         .WORD   SYSUAFALT_LEN,LNM$_STRING
                   00000414'         046B    756         .LONG   SYSUAFALT
        00000000 00000000            046F    757         .QUAD   0
                                     0477    758
```

```
                                M 16
0477    760          .SBTTL  SYSTEM INITIALIZATION PROCESS
0477    761 ;++
0477    762 ; FUNCTIONAL DESCRIPTION:
0477    763 ;
0477    764 ;       THIS PROCESS IS INITIATED BY THE OPERATING SYSTEM AFTER
0477    765 ;       IT HAS BEEN BOOT STRAPPED AND PROCESSOR INITIALIZTION
0477    766 ;       HAS BEEN COMPLETED. THE FOLLOWING FUNCTIONS ARE
0477    767 ;       PERFORMED:
0477    768 ;
0477    769 ;               1) THE PER-SYSTEM ROOT LOCK IS CREATED
0477    770 ;               2) CLUSTER INITIALIZATION
0477    771 ;                  IF NO CLUSTER:
0477    772 ;                       ENABLE UNCONSTRAINED LOCKING
0477    773 ;                  IF CLUSTER:
0477    774 ;                       STALL ROOT LOCK REQUESTS
0477    775 ;                       CREATE STAND-ALONE CONFIGURE PROCESS
0477    776 ;                       WAIT FOR CLUSTER TO FORM
0477    777 ;               3) SYSTEM LOGICAL NAMES ARE CREATED
0477    778 ;               4) PAGEFILE, SWAPFILE, AND RMS ARE INITIALIZED
0477    779 ;               5  MERGE FILE SYSTEM XQP.
0477    780 ;               6) THE SYSTEM DISK IS MOUNTED (ACP STARTED UP)
0477    781 ;               7) THE SYSTEM MESSAGE FILE IS OPENED AND MAPPED
0477    782 ;               8) STARTUP PROCESS IS INITIATED, WHICH NOW STARTS UP
0477    783 ;                  JOBCTL, OPCOM, AND ERRFMT.
0477    784 ;
0477    785 ;
0477    786 ; CALLING SEQUENCE:
0477    787 ;
0477    788 ;       NONE-ENTERED DIRECTLY FROM THE IMAGE ACTIVATOR
0477    789 ;
0477    790 ; INPUT PARAMETERS:
0477    791 ;
0477    792 ;       NONE
0477    793 ;
0477    794 ; IMPLICIT INPUTS:
0477    795 ;
0477    796 ;       LOGICAL NAME "SYS$SYSDEVICE" IS ASSIGNED TO THE SYSTEM DISK
0477    797 ;       FIL$GQ_CACHE CONTAINS A DESCRIPTOR FOR THE FIL$OPENFILE CACHE
0477    798 ;
0477    799 ; OUTPUT PARAMETERS:
0477    800 ;
0477    801 ;       NONE
0477    802 ;
0477    803 ; IMPLICIT OUTPUTS:
0477    804 ;
0477    805 ;       FILE ADDRESS ARE STORED, THE SPECIFIED PROCESSES ARE CREATED
0477    806 ;
0477    807 ;
0477    808 ; COMPLETION CODES:
0477    809 ;
0477    810 ;
0477    811 ; SIDE EFFECTS:
0477    812 ;
0477    813 ;       NONE
0477    814 ;
0477    815 ;--
0477    816          PURE_SECT
```

```
                                                B 1
                     0477   817
                     0477   818 SIP_START:
                0000 0477   819         .WORD   0                       ; ENTRY MASK
                     0479   820         $CMKRNL_S -
                     0479   821                 W^SIP_GET_SYSID_LOCK    ; OBTAIN LOCK FOR SYSTEM ID NAME
                     0486   822
                     0486   823         $CMKRNL_S W^SIP_SETTIME         ; SET THE INTERNAL SYSTEM TIME
                     0493   824
                     0493   825         $CMKRNL_S -
                     0493   826                 W^SIP_CLUSTER_INIT      ; CLUSTER RELATED INITIALIZATION
                     04A0   827
0000'CF   00   FB    04A0   828         CALLS   #0,W^LOCKDOWN           ; LOCK PAGES THAT MUST BE LOCKED
                     04A5   829
                     04A5   830 ;
                     04A5   831 ; CREATE THE SYSTEM LOGICAL NAMES. AN ASSUMPTION MADE IS THAT AN INDEX 0
                     04A5   832 ; TRANSLATION EXISTS FOR SYS$SYSDEVICE IF THE LOGICAL NAME IS SUCCESSFULLY
                     04A5   833 ; TRANSLATED.
                     04A5   834 ;
                     04A5   835
                     04A5   836         $TRNLNM_S -                     ; GET TRANSLATION ATTR OF THE SYSTEM DISK
                     04A5   837                 ITMLST = SYS_SYSDEVICE_ITMLST,-
                     04A5   838                 LOGNAM = SYS_SYSDEVICE_DESC,-
                     04A5   839                 TABNAM = LNM_FILE_DEV
      7E 50    E9    04BE   840         BLBC    R0,5$                   ; QUIT ON FAILURE
                     04C1   841
FFFFFCFF 8F    CA    04C1   842         BICL2   #^C<LNM$M_TERMINAL!LNM$M_CONCEALED>,-
000005BD'EF          04C7   843                 SYS_SYSDEVICE_ATTR      ; CLEAR UN-NEEDED ATTRIBUTES
                     04CC   844
                     04CC   845         $GETDVIW_S -                    ; GET FULL DEVICENAME OF THE SYSTEM DISK
                     04CC   846                 EFN = #1,-              ;
                     04CC   847                 IOSB = W^SIP_G_STATBLK,-
                     04CC   848                 ITMLST = SYS_SYSDEVICE_DVI_LST,-
                     04CC   849                 DEVNAM = SYS_SYSDEVICE_DESC
      52 50    E9    04EA   850         BLBC    R0,5$                   ; QUIT ON FAILURE
                     04ED   851
000005B1'FF   5F 8F  91    04ED   852         CMPB    #^A\_\,@SYS_SYSDEVICE_DEV
                0C   12    04F5   853         BNEQ    2$
      000005B1'EF    D6    04F7   854         INCL    SYS_SYSDEVICE_DEV       ; DISCARD LEADING "_"
      000005AD'EF    B7    04FD   855         DECW    SYS_SYSDEVICE_DEV_LEN
                     0503   856
                     0503   857 2$:     $CRELNM_S -                     ; SET UPTODATE TRANSLATION OF THE SYSTEM DIS
                     0503   858                 ITMLST = SYS_SYSDEVICE_ITMLST,-
                     0503   859                 LOGNAM = SYS_SYSDEVICE_DESC,-
                     0503   860                 ACMODE = EXEC_MODE,-    ;
                     0503   861                 TABNAM = LNM_SYSTEM_DESC
      1E 50    E9    051E   862         BLBC    R0,5$                   ; QUIT ON FAILURE
                     0521   863
                     0521   864         $CRELNM_S -                     ; SET UPTODATE TRANSLATION OF THE SYSTEM DIS
                     0521   865                 ITMLST = SYS_SYSDEVICE_ITMLST,-
                     0521   866                 LOGNAM = SYS_DISK_DESC,-
                     0521   867                 ACMODE = EXEC_MODE,-    ;
                     0521   868                 TABNAM = LNM_SYSTEM_DESC
                     053C   869
      03 50    E8    053C   870         BLBS    R0,10$                  ; CONTINUE IF TRANSLATION EXISTS
       00F0    31    053F   871 5$:      BRW     CRELNM_FATAL            ; ELSE GENERATE ERROR
                     0542   872
                     0542   873 ;
```

```
                                0542   874 ; CREATE LOGICAL NAMES FOR SYS$COMMON AND SYS$SYSROOT.
                                0542   875 ;
                                0542   876
56   000005AD'EF    3C          0542   877 10$:    MOVZWL  SYS_SYSDEVICE_DEV_LEN,R6; SIZE OF DEVICE NAME TRANSLATION
57   000005B1'EF    D0          0549   878         MOVL    SYS_SYSDEVICE_DEV,R7    ; ADDRESS OF DEVICE NAME TRANSLATION
     53   57   56   C1          0550   879         ADDL3   R6,R7,R3                ; ADDRESS OF FIRST BYTE BEYOND DEVICE
                                0554   880                                         ; NAME TRANSLATION
                                0554   881
51   00000000'EF    DE          0554   882 20$:    MOVAL   FIL$GT_TOPSYS,R1        ; TOP LEVEL SYSTEM DIRECTORY IF ANY
          50   81   9A          0559   883         MOVZBL  (R1)+,R0                ; GET SIZE OF STRING
               0D   13          055E   884         BEQL    30$                     ; BRANCH IF NO TOP LEVEL DIRECTORY
       83   5B   8F   90        0560   885         MOVB    #^A/[/,(R3)+            ; BEGIN DIRECTORY STRING
     63   61   50   28          0564   886         MOVC3   R0,(R1),(R3)            ; MOVE THE TOP LEVEL DIRECTORY NAME
     83   5D2E 8F   B0          0568   887         MOVW    #^A/.]/,(R3)+           ; AND THE SEPARATOR
                                056D   888
     56   53   57   C3          056D   889 30$:    SUBL3   R7,R3,R6                ; GET SIZE OF THE EQUIVALENCE NAME
000005DD'EF   56   B0          0571   890         MOVW    R6,SYS_SYSROOT_TOPSYS_LEN; STORE THE LENGTH IN THE ITEM LIST
000005E1'EF   57   D0          0578   891         MOVL    R7,SYS_SYSROOT_TOPSYS   ; STORE THE ADDRESS IN THE ITEM LIST
                                057F   892         $CRELNM_S -                     ; CREATE SYS$SYSROOT LOGICAL NAME
                                057F   893                 ACMODE = EXEC_MODE,-    ;
                                057F   894                 ITMLST = SYS_SYSROOT_ITMLST,-
                                057F   895                 LOGNAM = SYS_SYSROOT_DESC,-
                                057F   896                 TABNAM = LNM_SYSTEM_DESC
          A2 50   E9            059A   897         BLBC    R0,5$                   ; GENERATE ERROR MESSAGE ON FAILURES
                                059D   898
     53   57   56   C1          059D   899         ADDL3   R6,R7,R3                ; ADDRESS OF FIRST BYTE BEYOND
                                05A1   900                                         ; SYS$SYSROOT CONSTRUCTED EQUIVALENCE
     51   FD64 CF   DE          05A1   901         MOVAL   CMNSYS,R1               ; COMMON SYSTEM ROOT IF ANY
          50   81   9A          05A6   902         MOVZBL  (R1)+,R0                ; GET SIZE OF STRING
 FF  A3   61   50   28          05A9   903         MOVC3   R0,(R1),-1(R3)          ; COPY THE COMMON SYSTEM ROOT NAME
     56   53   57   C3          05AE   904         SUBL3   R7,R3,R6                ; GET SIZE OF EQUIVALENCE NAME
00000591'EF   56   B0          05B2   905         MOVW    R6,SYS_SYSROOT_CMNSYS_LEN; SET EQUIVALENCE NAME SIZE IN ITEM LIST
00000595'EF   57   D0          05B9   906         MOVL    R7,SYS_SYSROOT_CMNSYS   ; SET EQUIVALENCE NAME ADDR IN ITEM LIST
                                05C0   907         $CRELNM_S -                     ; CREATE SYS$COMMON LOGICAL NAME
                                05C0   908                 ACMODE = EXEC_MODE,-
                                05C0   909                 ITMLST = SYS_COMMON_ITMLST,-
                                05C0   910                 LOGNAM = SYS_COMMON_DESC,-
                                05C0   911                 TABNAM = LNM_SYSTEM_DESC
          54 50   E9            05DB   912         BLBC    R0,CRELNM_FATAL         ; GENERATE ERROR MESSAGE ON FAILURES
                                05DE   913
                                05DE   914 ;
                                05DE   915 ; CREATE LOGICAL NAMES FOR SYS$MESSAGE, SYS$SHARE, AND SYS$SYSTEM.
                                05DE   916 ;
                                05DE   917
                                05DE   918         $CRELNM_S -                     ; CREATE SYS$MESSAGE LOGICAL NAME
                                05DE   919                 ACMODE = EXEC_MODE, -
                                05DE   920                 ITMLST = SYS_MESSAGE_ITMLST,-
                                05DE   921                 LOGNAM = SYS_MESSAGE_DESC,-
                                05DE   922                 TABNAM = LNM_SYSTEM_DESC
          38 50   E9            05F7   923         BLBC    R0,CRELNM_FATAL         ; GENERATE ERROR MESSAGE ON FAILURES
                                05FA   924
                                05FA   925         $CRELNM_S -                     ; CREATE SYS$SHARE LOGICAL NAME
                                05FA   926                 ACMODE = EXEC_MODE, -
                                05FA   927                 ITMLST = SYS_SHARE_ITMLST,-
                                05FA   928                 LOGNAM = SYS_SHARE_DESC,-
                                05FA   929                 TABNAM = LNM_SYSTEM_DESC
          1C 50   E9            0613   930         BLBC    R0,CRELNM_FATAL         ; GENERATE ERROR MESSAGE ON FAILURES
```

```
                                                D 1
                     0616   931
                     0616   932          $CRELNM_S -                     ; CREATE SYS$SYSTEM LOGICAL NAME
                     0616   933                  ACMODE = EXEC_MODE, -
                     0616   934                  ITMLST = SYS_SYSTEM_ITMLST,-
                     0616   935                  LOGNAM = SYS_SYSTEM_DESC,-
                     0616   936                  TABNAM = LNM_SYSTEM_DESC
           08 50  E8 062F   937          BLBS    R0,CRELNM_DONE          ; GENERATE ERROR MESSAGE ON FAILURES
                     0632   938
                     0632   939 ;
                     0632   940 ; FAILED TO CREATE THE SYSTEM LOGICAL NAMES.
                     0632   941 ;
                     0632   942
                     0632   943 CRELNM_FATAL:
        51  FA5F CF  DE 0632   944          MOVAL   W^CRELNMERR,R1          ; ERROR MESSAGE TEXT
             0BBF  30 0637   945          BSBW    SIP_FATAL               ; REPORT ERROR AND QUIT
                     063A   946
                     063A   947 ;
                     063A   948 ; SUCCESSFULLY CREATED THE SYSTEM LOGICAL NAMES.
                     063A   949 ;
                     063A   950
                     063A   951 CRELNM_DONE:                             ; SUCCESSFULLY CREATED LOGICAL NAMES
17 00000000'EF  00000000'8F  E1 063A   952          BBC     #EXE$V_SYSUAFALT,EXE$GL_FLAGS,10$ ; BR IF NORMAL NAME FOR SYSUAF
                     0646   953          $CRELNM_S -                     ; EQUATE SYSUAF TO ALTERNATE NAME
                     0646   954                  ITMLST = W^SYSUAF_ITMLST,-
                     0646   955                  LOGNAM = W^SYSUAF_DESC,-
                     0646   956                  TABNAM = W^LNM_SYSTEM_DESC
                     065D   957
                     065D   958 ;
                     065D   959 ; THE FILE SYSTEM AND RMS ARE NOT YET AVAILABLE, USE THE BOOTSTRAP
                     065D   960 ; FIL$OPENFILE CODE TO "OPEN" THE FILES THAT MUST BE PRESENT BEFORE
                     065D   961 ; THE FILE SYSTEM CAN BE INITIALIZED.
                     065D   962 ;
              56  7C 065D   963 10$:     CLRQ    R6                      ; R6 = SIZE OF ATTRIBUTE REGION
                     065F   964                                          ; R7 = ADDRESS OF ATTRIBUTE REGION
        58  0128'CF  DE 065F   965          MOVAL   W^SIP_A_FILATT,R8       ; ARRAY OF FILE ATTRIBUTE POINTERS
        59  FC7B CF  DE 0664   966          MOVAL   W^SIP_A_NAMES,R9        ; ARRAY OF FILE NAME POINTERS
03 00000000'EF  00'  E1 0669   967          BBC     S^#EXE$V_PAGFILDMP,EXE$GL_FLAGS,30$ ; BRANCH IF DUMP
                     0671   968                                          ; IS NOT IN PAGE FILE
           89  88  D1 0671   969          CMPL    (R8)+,(R9)+             ; SYSBOOT "OPENED" PAGEFILE.SYS
                     0674   970                                          ; DON'T BOTHER DOING IT AGAIN
           51  89  D0 0674   971 30$:     MOVL    (R9)+,R1                ; ADR OF ASCIC FILE NAME STRING
               03  12 0677   972          BNEQ    32$                     ; PROCESS IT
                00A8  31 0679   973          BRW     50$                     ; BRANCH IF THIS IS THE END
           50  81  9A 067C   974 32$:     MOVZBL  (R1)+,R0                ; SIZE IN R0, ADR IN R1
      00E8'CF  50  7D 067F   975          MOVQ    R0,W^SIP_Q_TMPDESC      ; STORE FILE NAME DESCRIPTOR
           1C  56  D1 0684   976          CMPL    R6,#RTRVPTRS+8          ; ENOUGH ROOM IN RTRV BUFFER
                     0687   977                                          ; FOR FILE ATTRIBUTES AND AT LEAST
                     0687   978                                          ; ONE RETRIEVAL POINTER?
               2D  18 0687   979          BGEQ    36$                     ; BRANCH IF YES
                     0689   980 ;
                     0689   981 ; NEED TO ALLOCATE (MORE) SPACE FOR FILE ATTRIBUTES
                     0689   982 ;
        52  00E4'CF  01  C1 0689   983 34$:     ADDL3   #1,W^SIP_Q_RETADR+4,R2  ; FIRST ADDRESS OF NEXT PAGE TO
                     068F   984                                          ; BE EXPANDED INTO.  1 IF NO
                     068F   985                                          ; RTRV PTR BUFFER ALLOCATED YET.
                     068F   986          $EXPREG_S -
                     068F   987                  REGION=#0 -             ; GET THE NEXT PAGE IN P0 SPACE
```

```
                                068F   988                     PAGCNT=#1 -                     ; 1 PAGE
                                068F   989                     RETADR=W^SIP_Q_RETADR           ; RETURN ADDRESS RANGE
          56   0200 C6   DE    06A0   990             MOVAL    512(R6),R6                      ; FILE ATTRIBUTES BUFFER IS NOW
                                06A5   991                                                     ; 1 PAGE BIGGER (ASSUMING IT WAS
                                06A5   992                                                     ; ALLOCATED ADJACENT TO THE CURRENT BUF)
                                06A5   993
          52   00E0'CF   D1    06A5   994             CMPL     W^SIP_Q_RETADR,R2               ; DID WE ALLOCATE THE ADJACENT PAGE?
                     0A   13    06AA   995             BEQL     36$                             ; BRANCH IF YES
          57   00E0'CF   D0    06AC   996             MOVL     W^SIP_Q_RETADR,R7               ; SET NEW STARTING ADDRESS
          56   0200 8F   3C    06B1   997             MOVZWL   #512,R6                         ; AND SIZE FOR FILE ATTRIBUTES BUFFER
  00F8'CF   56   14   C3    06B6   998 36$:          SUBL3    #RTRVPTRS,R6,W^SIP_Q_RTRVBUF   ; SET UP SIZE AND ADDRESS
  00FC'CF   57   14   C1    06BC   999             ADDL3    #RTRVPTRS,R7,W^SIP_Q_RTRVBUF+4  ; OF RTRV PTR BUFFER
                                06C2  1000             $DASSGN_S W^SIP_L_DSKCHAN               ; FIL$OPENFILE ASSIGN CHANNEL EACH CALL
                                06CE  1001                                                     ; LEAVE IT ASSIGNED AFTER LAST CALL
00000000'EF   0104'CF   FA    06CE  1002             CALLG    W^SIP_A_OPENARG,FIL$OPENFILE    ; GET RETRIEVAL POINTERS
                                06D7  1003                                                     ; FOR SPECIFIED FILE
            1D 50   E8    06D7  1004             BLBS     R0,40$                          ; BRANCH IF SUCCESSFUL
        0000'8F   50   B1    06DA  1005             CMPW     R0,#SS$_NOSUCHFILE
                     08   13    06DF  1006             BEQL     38$                             ; IGNORE NO SUCH FILE
          51   FAF3 CF   DE    06E1  1007             MOVAL    W^FILOPNERR,R1
                  0B1B   30    06E6  1008             BSBW     SIP_SYSMSG                      ; DISPLAY ERROR
               67   01   CE    06E9  1009 38$:          MNEGL    #1,FILELBN(R7)                  ; SET IMPOSSIBLE STARTING LBN
                 04 A7   D4    06EC  1010             CLRL     FILESIZE(R7)                    ; SET SIZE=0
                 10 A7   D4    06EF  1011             CLRL     RTRVLEN(R7)                     ; NO RETRIEVAL POINTERS
               50   14   D0    06F2  1012             MOVL     #RTRVPTRS,R0                    ; NO. OF BYTE USED FOR ATTRIBUTES
                    18   11    06F5  1013             BRB      44$
                                06F7  1014 ;
                                06F7  1015 ; SUCCESS RETURN FROM FIL$OPENFILE
                                06F7  1016 ;
          67   00F0'CF   7D    06F7  1017 40$:          MOVQ     W^SIP_Q_STATBLK,STATBLK(R7) ; STORE STATISTICS BLOCK
    10 A7   0100'CF   D0    06FC  1018             MOVL     W^SIP_L_RTRVLEN,RTRVLEN(R7) ; AND RTRV PTR BYTE COUNT
    50   10 A7   14   C1    0702  1019             ADDL3    #RTRVPTRS,RTRVLEN(R7),R0 ; FORM BYTE COUNT USED IF ALL
                                0707  1020                                                     ; THE RETRIEVAL POINTERS FIT IN
                                0707  1021                                                     ; THE SPECIFIED BUFFER SPACE.
               56   50   D1    0707  1022             CMPL     R0,R6                           ; WAS THERE ENOUGH SPACE?
                    03   15    070A  1023             BLEQ     44$                             ; BRANCH IF NOT, GET MORE SPACE
                       FF7A   31    070C  1024             BRW      34$                             ; AND TRY THE FIL$OPENFILE AGAIN
                                070F  1025 ;
                                070F  1026 ; R0 = THE NUMBER OF BYTE USED FOR THE FILE ATTRIBUTES FOR THIS FILE
                                070F  1027 ;
          08 A7   01   D0    070F  1028 44$:          MOVL     #1,IMAGEVBN(R7)                 ; INIT IMAGE ATTRIBUTES
    0C A7   04 A7   D0    0713  1029             MOVL     FILESIZE(R7),IMAGESIZE(R7) ; AS IF NOT AN IMAGE FILE
               88   57   D0    0718  1030             MOVL     R7,(R8)+                        ; STORE THE POINTER TO THE
                                071B  1031                                                     ; ATTRIBUTES FOR THIS FILE
               57   50   C0    071B  1032             ADDL     R0,R7                           ; UPDATE BUFFER ADDRESS
               56   50   C2    071E  1033             SUBL     R0,R6                           ; AND SIZE
                       FF50   31    0721  1034             BRW      30$                             ; GO PROCESS THE NEXT FILE
                                0724  1035
          57   0130'CF   D0    0724  1036 50$:          MOVL     W^SIP_L_RMSATT,R7               ; RMS FILE ATTRIBUTES
          50   0134'CF   3C    0729  1037             MOVZWL   W^SIP_L_DSKCHAN,R0              ; CHANNEL TO READ FROM
            51   18 A7   D0    072E  1038             MOVL     RTRVPTRS+4(R7),R1               ; LBN OF FIRST BLOCK OF FILE
               53   21   D0    0732  1039             MOVL     #IO$_READLBLK,R3                ; FUNCTION CODE
            52   0C A7   D0    0735  1040             MOVL     IMAGESIZE(R7),R2                ; ACTUAL LAST VBN IN FILE
                  060D   30    0739  1041             BSBW     SIP_IMAGE_ATT                   ; GET IMAGE ATTRIBUTES
                 0A 50   E9    073C  1042             BLBC     R0,52$                          ; BRANCH IF ERROR
08 A7   51   01   C1    073F  1043             ADDL3    #1,R1,IMAGEVBN(R7)              ; SAVE STARTING VBN OF IMAGE
0C A7   52   51   C3    0744  1044             SUBL3    R1,R2,IMAGESIZE(R7)             ; SAVE BLOCKS OF IMAGE TO MAP
```

```
                                                F 1
                     0749  1045 52$:    $CMKRNL_S       W^SIP_KERNELRTN ; EXECUTE THIS AT KERNEL ACCESS MODE
                     0756  1046         $CMEXEC_S       SIP_XQP_MERGE
           08 50  E8 0765  1047         BLBS    R0,60$
      51 FAA9 CF  9E 0768  1048         MOVAB   W^XQPERR,R1
            0A94  30 076D  1049         BSBW    SIP_SYSMSG
      7E 0134'CF  3C 0770  1050 60$:    MOVZWL  W^SIP_L_DSKCHAN,-(SP)   ; GET CHANNEL ASSIGNED TO SYSTEM DISK
              01  DD 0775  1051         PUSHL   #1                      ; BUILD ARG LIST
              5E  DD 0777  1052         PUSHL   SP                      ; FOR $CMEXEC CALL
      00000000'EF 9F 0779  1053         PUSHAB  MOUNT_SYSTEM            ; SYSTEM DISK MOUNT ROUTINE
00000000'GF   04  FB 077F  1054         CALLS   #4,G^SYS$CMEXEC         ; GO MOUNT SYSTEM DISK
           08 50  E8 0786  1055         BLBS    R0,65$                  ; BR IF MOUNT WENT OK
      51 F99E CF  9E 0789  1056         MOVAB   W^MOUERR,R1             ; SET ERROR MESSAGE
            0A73  30 078E  1057         BSBW    SIP_SYSMSG              ; OUTPUT SYSTEM MESSAGE
                     0791  1058 65$:
                     0791  1059 ;
                     0791  1060 ; STORE THE SYSTEM TIME AND THE SYSGEN PARAMETERS IN THE SYSTEM IMAGE
                     0791  1061 ; ON THE SYSTEM DISK.  THIS IS DONE AFTER THE SYSTEM DISK IS MOUNTED IN
                     0791  1062 ; ORDER TO AVOID WRITING TO THE DISK PRIOR TO MOUNTING IT.
                     0791  1063 ;
                     0791  1064         $SETIME_S                       ; UPDATE TIME AND SYSGEN PARAMETERS
                     079A  1065 ;
                     079A  1066 ; DEALLOCATE THE FIL$OPENFILE CACHE, WE NOW HAVE THE FILE SYSTEM UP
                     079A  1067 ;
                     079A  1068         $CMKRNL_S W^SIP_CACHE_DALC      ; DONE WITH FIL$OPENFILE CACHE
                     07A7  1069 ;
                     07A7  1070 ; IF THERE IS A TOP LEVEL SYSTEM DIRECTORY, ASK THE FILES ACP FOR ITS
                     07A7  1071 ; REAL NAME SO THAT THIS NAME WILL APPEAR IN THE SYSTEM WIDE LOGICAL
                     07A7  1072 ; NAMES RATHER THAN "SYSX".
                     07A7  1073 ;
                     07A7  1074 SIP_GET_TOPSYS:
   51 00000000'EF    9E 07A7  1075         MOVAB   FIL$GT_TOPSYS,R1        ; TOP LEVEL SYSTEM DIRECTORY STRING
         56    81    9A 07AE  1076         MOVZBL  (R1)+,R6                ; SIZE OF STRING IF PRESENT
               03    12 07B1  1077         BNEQ    5$                      ; BRANCH IF NO TOP LEVEL DIR
             0087    31 07B3  1078         BRW     20$
                     07B6  1079
   58   0000'CF    9E 07B6  1080 5$:     MOVAB   W^SIP_A_ERLBUFFER,R8    ; FORM ADDRESSES FOR 2
     57    56 AB    9E 07BB  1081         MOVAB   ATR$S_ASCNAME(R8),R7    ; FILE NAME SCRATCH BUFFERS
   67    61    56    28 07BF  1082         MOVC3   R6,(R1),(R7)            ; FORM NAME OF DIRECTORY TO LOOK UP
83  5249442E  8F    D0 07C3  1083         MOVL    #^A/.DIR/,(R3)+         ; TACK ON THE FILE TYPE
   83  313B  8F    B0 07CA  1084         MOVW    #^A/;1/,(R3)+           ; AND VERSION NUMBER
               67    9F 07CF  1085         PUSHAB  (R7)                    ; FORM DESCRIPTOR FOR DIR NAME
            06 A6    9F 07D1  1086         PUSHAB  6(R6)                   ; SIZE OF NAME + 6 CHARS
         50    5E    D0 07D4  1087         MOVL    SP,R0                   ; ADDRESS OF NAME DESCRIPTOR
                     07D7  1088         $QIOW_S -
                     07D7  1089                 CHAN=W^SIP_L_DSKCHAN -  ; CHANNEL
                     07D7  1090                 FUNC=#IO$_ACCESS -      ; FUNCTION CODE = ACCESS
                     07D7  1091                 EFN=#1 -                ; EVENT FLAG TO WAIT FOR
                     07D7  1092                 IOSB=W^SIP_Q_STATBLK -  ; I/O STATUS BLOCK
                     07D7  1093                 P1=W^SIP_Q_FIBDESC -    ; FILE ID BLOCK DESCRIPTOR
                     07D7  1094                 P2=R0 -                 ; FILE NAME DESCRIPTOR TO LOOK UP
                     07D7  1095                 P5=#SIP_A_ATRLIST       ; ATTRIBUTE LIST ADDRESS
         5E    08    C0 07FE  1096         ADDL    #8,SP                   ; CLEAN OFF NAME DESCRIPTOR
            14 50    E9 0801  1097         BLBC    R0,10$                  ; BRANCH IF I/O DID NOT GET QUEUED
      0F 00F0'CF    E9 0804  1098         BLBC    W^SIP_Q_STATBLK,10$     ; BRANCH IF I/O FAILED
68  0056 8F    2E    3A 0809  1099         LOCC    #^A/.7,#ATR$S_ASCNAME,(R8) ; FIND THE END OF THE DIR NAME
               07    13 080F  1100         BEQL    10$                     ; BRANCH IF NO NAME RETURNED
         56    51    58    C3 0811  1101         SUBL3   R8,R1,R6                ; GET SIZE OF NAME
```

```
                                                G 1
                57   58   D0  0815  1102                MOVL    R8,R7                           ; AND ADDRESS
 00000605'EF    56   B0  0818  1103 10$::              MOVW    R6,SYS_TOPSYS_DIRNAM_LEN        ; SET SIZE OF EQUIVALENCE NAME
 00000609'EF    57   D0  081F  1104                    MOVL    R7,SYS_TOPSYS_DIRNAM            ; SET ADDRESS OF EQUIVALENCE NAME
                         0826  1105                    $CRELNM_S -                             ; CREATE LOGICAL NAME FOR SYS$TOPSYS
                         0826  1106                            ITMLST = W^SYS_TOPSYS_ITMLST,-
                         0826  1107                            LOGNAM = W^SYS_TOPSYS_DESC,-
                         0826  1108                            TABNAM = W^LNM_SYSTEM_DESC
                         083D  1109
                         083D  1110 ;
                         083D  1111 ; OPEN AND CREATE GLOBAL SECTIONS FOR THE XQP
                         083D  1112 ;
                         083D  1113
                         083D  1114 20$:                $OPEN   FAB = W^XQPFAB                  ; OPEN IT
                   44 50  E9  0848  1115                BLBC    R0,40$                          ; ERROR OPENING FILE
                         084B  1116                    $QIOW_S CHAN = XQPFAB+FAB$L_STV,-       ; READ IMAGE HEADER
                         084B  1117                            FUNC = #IO$_READVBLK,-
                         084B  1118                            IOSB = W^SIP_Q_STATBLK,-
                         084B  1119                            P1   = W^SIP_A_ERLBUFFER,-
                         084B  1120                            P2   = #512,-
                         084B  1121                            P3   = #1
                   18 50  E9  0874  1122                BLBC    R0,40$                          ; ERROR READING FILE
              50   00F0'CF  3C  0877  1123                MOVZWL  W^SIP_Q_STATBLK,R0
                   10 50  E9  087C  1124                BLBC    R0,40$
                         087F  1125                    $CMKRNL_S W^SIP_MAPXQP                  ; GO MAP XQP IN KERNEL MODE
                   08 50  E8  088C  1126                BLBS    R0,50$
              51   F982 CF  9E  088F  1127 40$:         MOVAB   W^XQPERR,R1
                     096D  30  0894  1128                BSBW    SIP_SYSMSG
                         0897  1129 50$:
                         0897  1130 ;
                         0897  1131 ; NOW OPEN AND MAP THE SYSTEM WIDE MESSAGE FILE (SYS$MESSAGE:SYSMSG.EXE)
                         0897  1132 ;
                         0897  1133                    $OPEN   FAB=W^MSGFILFAB                 ; OPEN THE FILE
                   54 50  E9  08A2  1134                BLBC    R0,74$                          ; BRANCH IF ERROR
              50   000C'CF  3C  08A5  1135                MOVZWL  W^MSGFILFAB+FAB$L_STV,R0        ; CHANNEL TO READ FROM
                   51   01  D0  08AA  1136                MOVL    #1,R1                           ; READ VIRTUAL BLOCK 1
                   53   31  D0  08AD  1137                MOVL    #IO$_READVBLK,R3                ; FUNCTION CODE
           52   00000060'EF  D0  08B0  1138                MOVL    MSGFILXAB+XAB$L_EBK,R2          ; END OF FILE BLOCK NUMBER
                00000064'EF  B5  08B7  1139                TSTW    MSGFILXAB+XAB$W_FFB             ; UNLESS FIRST FREE BYTE = 0
                         02  12  08BD  1140                BNEQ    72$
                         52  D7  08BF  1141                DECL    R2                              ; IN WHICH CASE IT IS ONE TOO BIG
                     0485  30  08C1  1142 72$:             BSBW    SIP_IMAGE_ATT                   ; GET IMAGE ATTRIBUTES
                   32 50  E9  08C4  1143                BLBC    R0,74$                          ; BRANCH IF ERROR
                52   51  C2  08C7  1144                SUBL    R1,R2                           ; NUMBER OF BLOCKS TO ACTUALLY MAP
                         2D  15  08CA  1145                BLEQ    74$                             ; BRANCH IF NOTHING TO MAP
                         08CC  1146 ;
                         08CC  1147 ; MAP THE MESSAGE FILE AS A SYSTEM SECTION
                         08CC  1148 ;
                00000000'EF  DF  08CC  1149                PUSHAL  EXE$GL_SYSMSG                   ; LOCATION TO STORE SYSTEM
                         08D2  1150                                                             ; ADDRESS AT WHICH SYSMSG IS MAPPED
                         0F  DD  08D2  1151                PUSHL   #PRT$C_UR                       ; PROTECTION FOR PAGES
                         52  DD  08D4  1152                PUSHL   R2                              ; PAGE COUNT TO MAP
              7E   51   01  C1  08D6  1153                ADDL3   #1,R1,-(SP)                     ; STARTING VBN TO MAP
           7E   0000000C'EF  3C  08DA  1154                MOVZWL  MSGFILFAB+FAB$L_STV,-(SP)       ; CHANNEL ON WHICH SYSMSG IS OPEN
                         05  DD  08E1  1155                PUSHL   #5                              ; NO. OF ARGUMENTS IN THE ARG LIST
                   50   5E  D0  08E3  1156                MOVL    SP,R0                           ; ADDRESS OF ARGUMENT LIST
                         08E6  1157                    $CMKRNL_S W^EXE$SYS_SECTION,(R0)                ; MAP THE SECTION
                   5E   18  C0  08F3  1158                ADDL    #<6*4>,SP                       ; CLEAN OFF ARGUMENT LIST
```

```
                                                     H 1
                      08 50   E8  08F6  1159          BLBS    R0,90$                          ; BRANCH IF SUCCESSFUL
                51    F7DC CF   9E  08F9  1160 74$:   MOVAB   W^MSGFILERR,R1                  ; "FAILED TO OPEN OR MAP SYSMSG.EXE"
                        0903    30  08FE  1161          BSBW    SIP_SYSMSG                      ; ISSUE A WARNING DIAGNOSTICE
                                    0901  1162 90$:
        01EC'CF   00000000'EF   9A  0901  1163          MOVZBL  EXE$GT_STARTUP,W^SIP_Q_SPINPUT  ; SET CORRECT COUNT IN DESCR
                50    F721 CF   9E  090A  1164          MOVAB   W^SIP_Q_SPOUTPUT,R0             ; STARTUP PROCESS OUTPUT
              00     00000000'8F  D1  090F  1165          CMPL    #XDT$START,#0                   ; DEBUGGING WITH DELTA?
                          05    13  0916  1166          BEQL    95$                             ; BRANCH IF NOT
                50    F720 CF   9E  0918  1167          MOVAB   W^SIP_Q_SPOUTXDT,R0             ; USE DIFFERENT OUTPUT FOR DELTA
                                    091D  1168 95$:     $CREPRC_S       INPUT=W^SIP_Q_SPINPUT,- ; INPUT FROM STARTUP FILE
                                    091D  1169                          OUTPUT=(R0),-           ; OUTPUT TO CONSOLE TERMINAL
                                    091D  1170                          ERROR=(R0),-            ; ERRORS ALSO
                                    091D  1171                          BASPRI=#4,-             ; BASE PRIORITY
                                    091D  1172                          IMAGE=W^SIP_Q_SPIMAGE,- ; RUN LOGIN IMAGE
                                    091D  1173                          UIC=#^X10004,-          ; RUN IN UIC [1,4]
                                    091D  1174                          STSFLG=#<1@6>,-         ; FLAG FOR AUTO LOGIN
                                    091D  1175                          PRVADR=W^SIP_Q_PRVMSK,- ; ALL PRIVILEGES
                                    091D  1176                          QUOTA=PQL$AB_SYSPQL,-   ; QUOTA LIST
                                    091D  1177                          PRCNAM=W^SIP_Q_STARTUP  ; NAME IS STARTUP
                      47 50   E8  0953  1178          BLBS    R0,100$                         ; BR IF SUCCESS
                         3F   BB  0956  1179          PUSHR   #^M<R0,R1,R2,R3,R4,R5>          ; SAVE REGISTERS
                50    F6C4 CF   7E  0958  1180          MOVAQ   W^SIP_Q_STARTUP,R0              ; GET PROCESS NAME DESCRIPTOR
01DD'CF   0F    00      04 B0    60  2C  095D  1181          MOVC5   (R0),@4(R0),#0,#15,W^CREPRCNAM ; COPY NAME INTO MESSAGE
                51    01C4'CF   9E  0966  1182          MOVAB   W^CREPRCERR,R1                  ; SET ADDR OF MESSAGE
                      50    6E  D0  096B  1183          MOVL    (SP),R0                         ; SET FAILURE STATUS VALUE
                          0893  30  096E  1184          BSBW    SIP_SYSMSG                      ; PRINT THE MESSAGE
                            3F  BA  0971  1185          POPR    #^M<R0,R1,R2,R3,R4,R5>          ; RESTORE REGISTERS
                                    0973  1186          $GETMSG_S R0,SIP_Q_TMPDESC,SIP_Q_LINBUF ; GET STATUS MESSAGE
                50    000000E8'EF   3C  098C  1187          MOVZWL  SIP_Q_TMPDESC,R0                ; GET SIZE OF MESSAGE
                51    00000140'EF   9E  0993  1188          MOVAB   SIP_T_LINBUF,R1                 ; GET ADDR OF MESSAGE
                          088B  30  099A  1189          BSBW    SIP_TYPOUT                      ; TYPE IT ON CONSOLE
                                    099D  1190 100$:
                                04  099D  1191          RET                                     ; THATS ALL FOR NOW
```

```
                                                     I  1
                                .SUBTITLE       SIP_GET_SYSID_LOCK - Obtain Lock for System ID
                                ;+
                                ; Functional Description:
                                ;
                                ;       This routine obtains a system-owned lock whose name contains the
                                ;       system ID. If this system is to join a cluster, a test will be made
                                ;       for a unique system ID when this system's lock data base is merged
                                ;       into the cluster-wide data base. The lock is system wide because the
                                ;       various sublocks that will use this as a parent are locking system
                                ;       wide data structures.
                                ;
                                ;       If $ENQW returns an error, a message will be issued and the SYSINIT
                                ;       image will go away, preventing further system initialization
                                ;
                                ;       Locking is enabled before this lock is requested and sub-locks are
                                ;       enabled after the lock is granted.
                                ;
                                ; Calling Sequence:
                                ;
                                ;       CALLS   #0, SIP_GET_SYSID_LOCK
                                ;
                                ; Environment:
                                ;
                                ;       This routine must execute in kernel mode
                                ;
                                ; Input Parameters:
                                ;
                                ;       none
                                ;
                                ; Output Parameters:
                                ;
                                ;       none
                                ;
                                ; Implicit Output:
                                ;
                                ;       If the lock request is successful, the lock ID is stored in the
                                ;       exec cell called EXE$GL_SYSID_LOCK for use as a parent ID by other
                                ;       lock requests.
                                ;
                                ;       If the lock request fails, the image exits (and initialization
                                ;       terminates) after an error message is typed.
                                ;-
                                
                                SIP_GET_SYSID_LOCK:
                        0000    .WORD   0                                       ; Save no registers
                                
                                ; Enable locking
                                
                00000000'GF  94 CLRB    G^LCK$GB_STALLREQS
                                
                                ; Take out an exclusive lock with the system ID as the lock name
                                
0000045D'EF     00000000'GF  D0 MOVL    G^SCS$GB_SYSTEMID, SYS_ID       ; Move first four bytes of ID
00000461'EF     00000000'GF  B0 MOVW    G^SCS$GB_SYSTEMIDH, SYS_ID + 4  ;  and the last two bytes, too
                                
                                $ENQW_S         EFN = #32,-
                                                LKMODE = #LCK$K_EXMODE,-
```

```
                                09BC  1250                      LKSB = LOCK_STATUS_BLOCK,-
                                09BC  1251                      FLAGS = #LOCK_FLAGS,-
                                09BC  1252                      RESNAM = LOCK_NAME_DESC,-
                                09BC  1253                      ACMODE = #PSL$C_EXEC
                                09E3  1254
                      16 50     E9  09E3  1255          BLBC    R0, ERROR                       ; Abort image if an error occurs
                                09E6  1256
                                09E6  1257 ; Store the lock ID where other folks can find it and return success
                                09E6  1258
00000000'GF   0000044F'EF       D0  09E6  1259          MOVL    LOCK_ID, G^EXE$GL_SYSID_LOCK    ; Store the lock ID
                                09F1  1260
                                09F1  1261 ; Enable sub-locking, but not creation of additional roots
                                09F1  1262
          00000000'GF   02      90  09F1  1263          MOVB    #2,G^LCK$GB_STALLREQS
                                09F8  1264
                  50   00'     3C  09F8  1265          MOVZWL  S^#SS$_NORMAL,R0                ; Indicate success
                               04   09FB  1266          RET                                     ;  and return
                                09FC  1267
                                09FC  1268 ERROR:
                51   F836 CF     9E  09FC  1269          MOVAB   SYSID_LOCK_ERR, R1              ; Store error message address
                       07F5     31  0A01  1270          BRW     SIP_FATAL                       ; This is the death step
```

```
                     0A04  1272          .SUBTITLE       SIP_CLUSTER_INIT - Cluster related initialization
                     0A04  1273 ;+
                     0A04  1274 ; Functional Description:
                     0A04  1275 ;
                     0A04  1276 ;       This routine performs cluster related initializations.
                     0A04  1277 ;
                     0A04  1278 ;       If the node is not even going to participate in a cluster, locking
                     0A04  1279 ;       is enabled and the routine returns.
                     0A04  1280 ;
                     0A04  1281 ;       If the node will participate in a cluster:
                     0A04  1282 ;
                     0A04  1283 ;       1.  The stand-alone configure process is created.  The purpose of
                     0A04  1284 ;           this process is to configure communications drivers supporting
                     0A04  1285 ;           SCS and the disk driver supporting the disk potentially
                     0A04  1286 ;           containing the quorum file.
                     0A04  1287 ;
                     0A04  1288 ;       2.  A bit is set triggering cluster formation/joining.
                     0A04  1289 ;
                     0A04  1290 ;       3.  Wait for a cluster to be joined or formed.  It is assumed that
                     0A04  1291 ;           locking is enabled as a side effect of joining or forming the
                     0A04  1292 ;           cluster.
                     0A04  1293 ;
                     0A04  1294 ;       4.  Time is updated to set a consistent, cluster-wide time.
                     0A04  1295 ;
                     0A04  1296 ;
                     0A04  1297 ; Calling Sequence:
                     0A04  1298 ;
                     0A04  1299 ;       CALLS   #0, SIP_CLUSTER_INIT
                     0A04  1300 ;
                     0A04  1301 ; Environment:
                     0A04  1302 ;
                     0A04  1303 ;       This routine must execute in kernel mode
                     0A04  1304 ;
                     0A04  1305 ; Input Parameters:
                     0A04  1306 ;
                     0A04  1307 ;       none
                     0A04  1308 ;
                     0A04  1309 ; Output Parameters:
                     0A04  1310 ;
                     0A04  1311 ;       none
                     0A04  1312 ;
                     0A04  1313 ; Implicit Output:
                     0A04  1314 ;
                     0A04  1315 ;-
                     0A04  1316
                     0A04  1317 SIP_CLUSTER_INIT:
                003C 0A04  1318          .WORD   ^M<R2,R3,R4,R5>
                     0A06  1319
                     0A06  1320          IFCLSTR 2$                                  ; Branch if cluster system
                     0A0E  1321 ;
                     0A0E  1322 ; This system will never participate in a cluster; enable unrestricted locking
                     0A0E  1323 ;
00000000'GF    94    0A0E  1324          CLRB    G^LCK$GB_STALLREQS
          00E8 31    0A14  1325          BRW     30$
                     0A17  1326
                     0A17  1327 ; Create the stand-alone configure process
                     0A17  1328 ;
```

```
                     0A17  1329 2$:     $CREPRC_S       IMAGE = W^STAC_IMAGE,-
                     0A17  1330                     INPUT = W^STAC_OPER,-
                     0A17  1331                     OUTPUT = W^STAC_OPER,-
                     0A17  1332                     ERROR = W^STAC_OPER,-
                     0A17  1333                     PRVADR = W^STAC_PRV_MSK,-
                     0A17  1334                     QUOTA = W^STAC_QLIST,-
                     0A17  1335                     PRCNAM = W^STAC_PRC,-
                     0A17  1336                     BASPRI = #8,-
                     0A17  1337                     UIC = #^x10004
           03 50 E8  0A4B  1338             BLBS    R0,3$                   ; Branch on success
            00BA 31  0A4E  1339             BRW     100$                    ; Can't create process
                     0A51  1340 3$:
                     0A51  1341
                     0A51  1342 ; Tell the connection manager to proceed with cluster formation/creation
                     0A51  1343 ;
      50 00000000'GF D0  0A51  1344             MOVL    G^CLU$GL_CLUB,R0        ; Address of CLUster Block
   1C A0 00080000 8F C8  0A58  1345             BISL2   #CLUB$M_INIT,CLUB$L_FLAGS(R0)   ; Set initialization flag
                     0A60  1346
                     0A60  1347 ; Output message indicating that we are waiting to join/form a cluster
                     0A60  1348 ;
         51 F7FF CF  9E  0A60  1349             MOVAB   W^SIP_CLU_MSG,R1        ; Address of counted string
            50 81 9A  0A65  1350             MOVZBL  (R1)+,R0                ; Character count
               07BD 30  0A68  1351             BSBW    SIP_TYPOUT
                     0A6B  1352
                     0A6B  1353 ; Loop waiting for node to join cluster
                     0A6B  1354 ;
                     0A6B  1355 10$:     $SETIMR_S       EFN=#0,DAYTIM=W^SIP_CLU_TIMOUT
           09 50 E9  0A7A  1356             BLBC    R0,15$                  ; Branch on error
                     0A7D  1357             $WAITFR_S       EFN=#0          ; Wait for time-out
               009C 30  0A86  1358 15$:    BSBW    SIP_LOOKUP_QFILE        ; Perform quorum file lookup
   50 00000000'GF D0  0A89  1359             MOVL    G^CLU$GL_CLUB,R0        ; Address of CLUster Block
D6 1C A0 00 E1  0A90  1360             BBC     #CLUB$V_CLUSTER, -      ; Loop until node is a
                     0A95  1361                     CLUB$L_FLAGS(R0),10$    ;   cluster member
                     0A95  1362                                             ; If it was not already, start
               01AD 30  0A95  1363             BSBW    SIP_START_QUORUM_TIMER  ; ...the quorum disk timer
                     0A98  1364 ;
                     0A98  1365 ; When the cluster if formed or joined, the locking will be enabled --
                     0A98  1366 ; i.e., it is enabled when we reach this point.  Take out a lock on the
                     0A98  1367 ; system disk.
                     0A98  1368 ;
   54 00000000'GF D0  0A98  1369             MOVL    G^CTL$GL_PCB,R4         ; PCB address
      00000000'GF 16  0A9F  1370             JSB     G^SCH$IOLOCKW           ; Lock I/O data base for writing
         50 01 D0  0AA5  1371             MOVL    #LCK$K_CRMODE,R0        ; Signal shared lock
            51 D4  0AA8  1372             CLRL    R1                      ; Don't return lock status block
   55 00000000'GF D0  0AAA  1373             MOVL    G^EXE$GL_SYSUCB,R5      ; System disk UCB address
     3C A5 01 88  0AB1  1374             BISB2   #DEV$M_CLU, UCB$L_DEVCHAR2(R5)  ; It is cluster accessible.
      00000000'GF 16  0AB5  1375             JSB     G^IOC$LOCK_DEV          ; Take out lock on system disk
                     0ABB  1376 ;
                     0ABB  1377 ; The UCB for the system disk was created with a reference count of 1 to
                     0ABB  1378 ; avoid having the first $ASSIGN try to take out a lock on it before locking
                     0ABB  1379 ; is enabled. If SYSINIT fails for any reason (e.g. failure to mount the
                     0ABB  1380 ; system disk), this extra reference count will prevent the device lock from
                     0ABB  1381 ; being released in the last channel $DASSGN. Decrement the reference count
                     0ABB  1382 ; to avoid this scenario.
                     0ABB  1383 ;
           5C A5 B7  0ABB  1384             DECW    UCB$W_REFC(R5)          ; Decrement reference count
               50 DD  0ABE  1385             PUSHL   R0                      ; Save LOCK_DEV status
```

```
                                                M 1
                00000000'GF   16  0AC0  1386          JSB     G^SCH$IOUNLOCK                  ; Unlock the I/O data base
                              0AC6  1387          SETIPL  #0                              ; Restore IPL
                     50   BE  D0  0AC9  1388          MOVL    (SP)+,R0                        ; Retrieve LOCK_DEV status
                       34 50  E9  0ACC  1389          BLBC    R0,90$                          ; Branch if LOCK_DEV failed
                              0ACF  1390 ;
                              0ACF  1391 ; Set internal cluster-wide system time using data that was stored in the CLUB
                              0ACF  1392 ; when the cluster was formed/joined.
                              0ACF  1393 ;
             7E   00000000'GF  7D  0ACF  1394          MOVQ    G^EXE$GQ_SYSTIME,-(SP)          ; Current system time
             50   00000000'GF  D0  0AD6  1395          MOVL    G^CLU$GL_CLUB,R0                ; Address of CLUB
                6E    009C C0  C2  0ADD  1396          SUBL2   CLUB$Q_NEWTIME_REF(R0),(SP)     ; Subtract local time corresponding
             04 AE    00A0 C0  D9  0AE2  1397          SBWC    CLUB$Q_NEWTIME_REF+4(R0),4(SP)  ;  cluster time
                6E    0094 C0  C0  0AE8  1398          ADDL2   CLUB$Q_NEWTIME(R0),(SP)         ; Add cluster time corresponding to
             04 AE    0098 C0  D8  0AED  1399          ADWC    CLUB$Q_NEWTIME+4(R0),4(SP)      ;  reference base
                      6E   7F  0AF3  1400          PUSHAQ  (SP)                            ; Address of new system time
             00000000'GF   01  FB  0AF5  1401          CALLS   #1,G^EXE$SETIME_INT             ; Establish cluster-wide time intern
                      5E   08  C0  0AFC  1402          ADDL2   #8,SP                           ; Clear stack
                      50   00' 3C  0AFF  1403 30$:      MOVZWL  S^#SS$_NORMAL,R0                ; Indicate success
                           04  0B02  1404          RET                                     ;  and return
                               0B03  1405
                               0B03  1406 ;
                               0B03  1407 ; Error locking system disk - this is fatal.
                               0B03  1408 ;
                   51   F641 CF  9E  0B03  1409 90$:      MOVAB   W^LOCKERR,R1                    ; Message address
                          06EE  31  0B08  1410          BRW     SIP_FATAL                       ; No recovery possible
                               0B0B  1411
                               0B0B  1412 ; Error creating stand-alone configure process
                               0B0B  1413 ;
                         50    DD  0B0B  1414 100$:     PUSHL   R0                              ; Save failure status
                      50   0496'CF  7E  0B0D  1415          MOVAQ   W^STAC_PRC,R0                   ; Get process name descriptor
01DD'CF    0F    00    04 B0    60  2C  0B12  1416          MOVC5   (R0),@4(R0),#0,#15,W^CREPRCNAM  ; Copy name into message
                   51   01C4'CF  9E  0B1B  1417          MOVAB   W^CREPRCERR,R1                  ; Message address
                             01  BA  0B20  1418          POPR    #^M<R0>                         ; Failure status value
                           06D4  31  0B22  1419          BRW     SIP_FATAL                       ; This is the death step
```

```
                                                     N 1
                     0B25  1421          .SUBTITLE       SIP_LOOKUP_QFILE - Perform quorum file lookup
                     0B25  1422 ;+
                     0B25  1423 ; Functional Description:
                     0B25  1424 ;
                     0B25  1425 ;       This routine attempts to assign a channel to the quorum disk, get
                     0B25  1426 ;       the quorum file logical block number, and store it in the cluster
                     0B25  1427 ;       quorum disk control block (CLUDCB).
                     0B25  1428 ;
                     0B25  1429 ; Calling Sequence:
                     0B25  1430 ;
                     0B25  1431 ;       BSBW    SIP_LOOKUP_QFILE
                     0B25  1432 ;
                     0B25  1433 ; Environment:
                     0B25  1434 ;
                     0B25  1435 ;       This routine must execute in kernel mode
                     0B25  1436 ;
                     0B25  1437 ; Input Parameters:
                     0B25  1438 ;       none
                     0B25  1439 ;
                     0B25  1440 ; Output Parameters:
                     0B25  1441 ;       none
                     0B25  1442 ;-
                     0B25  1443
                     0B25  1444 SIP_LOOKUP_QFILE:
                     0B25  1445
     54  00000000'GF   D0  0B25  1446          MOVL    G^CLU$GL_CLUB,R4                ; Get CLUB address
         53  00B4 C4   D0  0B2C  1447          MOVL    CLUB$L_CLUDCB(R4),R3            ; Get CLUDCB address
                44   13  0B31  1448          BEQLU   1$                              ; If zero, there is no quorum file
              1C A3   D5  0B33  1449          TSTL    CLUDCB$L_QFLBN(R3)              ; Have we already found it?
                3F   12  0B36  1450          BNEQU   1$                              ; Br if yes
                     0B38  1451 ;
                     0B38  1452 ; Get the full device name, store it in the CLUB, and form the full quorum
                     0B38  1453 ; file specification.
                     0B38  1454 ;
             00B8 C4   95  0B38  1455          TSTB    CLUB$T_QDNAME(R4)               ; Is name already in CLUB?
                  73   12  0B3C  1456          BNEQU   3$                              ; Br if yes
                  18   BB  0B3E  1457          PUSHR   #^M<R3,R4>                      ; Save CLUDCB and CLUB pointers
                10    20   3A  0B40  1458          LOCC    #^A/ /,#CLUDCB$S_DISK_QUORUM,-  ; Locate end of quorum disk name
           00000000'GF       0B43  1459                  G^CLU$GB_QDISK
                10    50   A3  0B48  1460          SUBW3   R0,#CLUDCB$S_DISK_QUORUM,-      ; Adjust descriptor size
                04FD'CF       0B4B  1461                  W^SIP_QD_DESCR
                            0B4E  1462          $GETDVIW_S      EFN = #0,-              ; Get full device name
                            0B4E  1463                          DEVNAM = W^SIP_QD_DESCR,-
                            0B4E  1464                          ITMLST = W^SIP_QD_ITMLST,-
                            0B4E  1465                          IOSB = W^SIP_QD_IOSB
                   08 50    E9  0B6A  1466          BLBC    R0,7$                   ; Br if error
              50   04F5'CF   3C  0B6D  1467          MOVZWL  W^SIP_QD_IOSB,R0        ; Get completion status
                   05 50    E8  0B72  1468          BLBS    R0,2$                   ; Br if success
                   18   BA  0B75  1469  7$:     POPR    #^M<R3,R4>                      ; Restore registers
                   00CA 31  0B77  1470  1$:     BRW     6$
        50   0505'CF    02   A3  0B7A  1471  2$:     SUBW3   #2,W^SIP_QF_DESCR,R0            ; Get adjusted size
             0522'CF    50   28  0B80  1472          MOVC3   R0,W^SIP_QF_BUFFER+1,-          ; Put name in CLUB
                00B9 C4       0B85  1473                  CLUB$T_QDNAME+1(R4)
                   53   6E   7D  0B88  1474          MOVQ    (SP),R3                 ; Restore CLUDCB and CLUB pointers
              50   0505'CF   3C  0B8B  1475          MOVZWL  W^SIP_QF_DESCR,R0       ; Get size
        00B8 C4    50    02   83  0B90  1476          SUBB3   #2,R0,CLUB$T_QDNAME(R4) ; Put adjusted size in CLUB
    50   00000521'8F    50   C1  0B96  1477          ADDL3   R0,#SIP_QF_BUFFER,R0    ; Get address to put file name
```

```
                                                     B 2
                0505'CF    14   A0  0B9E  1478          ADDW    #SIP_QF_NAME_SIZE,W^SIP_QF_DESCR ; Add file name size into descr
                           14   28  0BA3  1479          MOVC3   #SIP_QF_NAME_SIZE,-              ; Move file name into buffer
                60    050D'CF                0BA5  1480                  W^SIP_QF_NAME,(R0)
                 00000000'GF    16  0BA9  1481          JSB     G^CNX$DISK_CHANGE                ; Tell connection manager
                           18   BA  0BAF  1482          POPR    #^M<R3,R4>                       ; Restore CLUDCB and CLUB pointers
                                    0BB1  1483 ;
                                    0BB1  1484 ; Assign a channel to the quorum disk and use the channel number to
                                    0BB1  1485 ; get the quorum disk UCB.
                                    0BB1  1486 ;
                52    04F1'CF    3E   0BB1  1487 3$:     MOVAW   W^SIP_QD_CHAN,R2                 ; R2 is channel word pointer
                  55     0C A3    D0  0BB6  1488          MOVL    CLUDCB$L_UCB(R3),R5              ; Get the quorum disk UCB address
                              24    12  0BBA  1489          BNEQU   4$                       ; Br if we have it
                                    0BBC  1490          $ASSIGN_S DEVNAM = W^SIP_QF_DESCR,-      ; Assign channel to quorum disk
                                    0BBC  1491                    CHAN = (R2)
                           76   50   E9  0BCB  1492          BLBC    R0,6$                    ; Br if error
                     55    62   3C   0BCE  1493          MOVZWL  (R2),R5                  ; Get channel number
          55    00000000'9F    55   C3  0BD1  1494          SUBL3   R5,@#CTL$GL_CCBBASE,R5           ; Form CCB address
                     55    65   D0   0BD9  1495          MOVL    CCB$L_UCB(R5),R5                 ; Get UCB address
                   0C A3    55   D0   0BDC  1496          MOVL    R5,CLUDCB$L_UCB(R3)              ; Store UCB address in CLUDCB
                                    0BE0  1497 ;
                                    0BE0  1498 ; The quorum disk may not be mounted. Check to see if the volume valid
                                    0BE0  1499 ; bit is set in the UCB.
                                    0BE0  1500 ;
                28 64 A5    0B   E0   0BE0  1501 4$:     BBS     #UCB$V_VALID,UCB$L_STS(R5),5$    ; Br if volume is valid
                                    0BE5  1502 ;
                                    0BE5  1503 ; The volume is not valid, issue a PACKACK QIO.
                                    0BE5  1504 ;
                                    0BE5  1505          $QIOW_S EFN  = #0,-                       ; Issue packack QIO
                                    0BE5  1506                  CHAN = (R2),-
                                    0BE5  1507                  FUNC = #IO$_PACKACK,-
                                    0BE5  1508                  IOSB = W^SIP_QD_IOSB
                     3F 50    E9   0C02  1509          BLBC    R0,6$                    ; Br if error on qio request
                50    04F5'CF    3C   0C05  1510          MOVZWL  W^SIP_QD_IOSB,R0                 ; Get I/O status
                     37 50    E9   0C0A  1511          BLBC    R0,6$                    ; Br if I/O error
                                    0C0D  1512 ;
                                    0C0D  1513 ; Do the file lookup with FILEREAD.
                                    0C0D  1514 ;
                        03    DD   0C0D  1515 5$:     PUSHL   #3                               ; Don't use cache or root directory
                        7E    7C   0C0F  1516          CLRQ    -(SP)                            ; We don't want retrieval pointers
                04F5'CF       DF   0C11  1517          PUSHAL  W^SIP_QD_STATBUF                 ; Address of 2 longword block to
                                    0C15  1518                                                   ; return LBN of the first block
                                    0C15  1519                                                   ; and the file size. (in blocks)
                0200'CF       DF   0C15  1520          PUSHAL  W^SIP_A_FILEHDR                  ; Address of file hdr buffer
                0000'CF       DF   0C19  1521          PUSHAL  W^SIP_A_INDEXFHDR                ; Address of index file hdr buffer
                0505'CF       DF   0C1D  1522          PUSHAL  W^SIP_QF_DESCR                   ; Address of file name descriptor
                        62    DF   0C21  1523          PUSHAL  (R2)                             ; Address of channel number
          00000000'GF    08   FB   0C23  1524          CALLS   #8,G^FIL$OPENFILE_1              ; Open quorum file
                     17 50    E9   0C2A  1525          BLBC    R0,6$                    ; Br if error
                                    0C2D  1526 ;
                                    0C2D  1527 ; We have found the quorum file. Store the logical block number in the CLUDCB.
                                    0C2D  1528 ;
                04F5'CF       D0   0C2D  1529          MOVL    W^SIP_QD_STATBUF,-               ; Store LBN in CLUDCB
                  1C A3            0C31  1530                  CLUDCB$L_QFLBN(R3)
                        02    B0   0C33  1531          MOVW    #CLUDCB$M_QS_READY,-             ; State is now READY
                  20 A3            0C35  1532                  CLUDCB$W_STATE(R3)
                   000B       30   0C37  1533          BSBW    SIP_START_QUORUM_TIMER           ; Start the quorum disk timer
                                    0C3A  1534          $DASSGN_S CHAN = (R2)                    ; Deassign channel
```

```
05  0C44  1535 6$:     RSB
```

```
                        0C45  1537          .SUBTITLE       SIP_START_QUORUM_TIMER - Start the quorum disk timer
                        0C45  1538 ;+
                        0C45  1539 ; Functional Description:
                        0C45  1540 ;
                        0C45  1541 ;       This routine starts the quorum disk timer by inserting the quorum
                        0C45  1542 ;       TQE in the system time queue. It first checks to see if it has
                        0C45  1543 ;       already been placed in the queue and if not requests an immediate
                        0C45  1544 ;       timeout.
                        0C45  1545 ;
                        0C45  1546 ; Calling Sequence:
                        0C45  1547 ;
                        0C45  1548 ;       BSBW    SIP_START_QUORUM_TIMER
                        0C45  1549 ;
                        0C45  1550 ; Environment:
                        0C45  1551 ;
                        0C45  1552 ;       This routine must execute in kernel mode
                        0C45  1553 ;
                        0C45  1554 ; Input Parameters:
                        0C45  1555 ;
                        0C45  1556 ;       none
                        0C45  1557 ;
                        0C45  1558 ; Output Parameters:
                        0C45  1559 ;
                        0C45  1560 ;       none
                        0C45  1561 ;-
                        0C45  1562
                        0C45  1563 SIP_START_QUORUM_TIMER:
     55  00000000'GF    DO  0C45  1564          MOVL    G^CLU$GL_CLUB,R5                ; Get CLUB address
         55    00B4 C5   DO  0C4C  1565          MOVL    CLUB$L_CLUDCB(R5),R5            ; Get CLUDCB address
               15        13  0C51  1566          BEQLU   1$                              ; If zero, there is no quorum file
           55    14 A5   DO  0C53  1567          MOVL    CLUDCB$L_TQE(R5),R5             ; Get TQE
               65        D5  0C57  1568          TSTL    TQE$L_TQFL(R5)                  ; Is it in queue already?
               0D        12  0C59  1569          BNEQU   1$                              ; Br if yes
     50  00000000'GF    7D  0C5B  1570          MOVQ    G^EXE$GQ_SYSTIME,R0             ; Request an immediate timeout
         00000000'GF    16  0C62  1571          JSB     G^EXE$INSTIMQ                   ; Insert in queue
                        05  0C68  1572 1$:      RSB
```

```
                        0C69  1574 ;++
                        0C69  1575 ; FUNCTIONAL DESCRIPTION:
                        0C69  1576 ;
                        0C69  1577 ;       Merge the XQP into this process.
                        0C69  1578 ;
                        0C69  1579 ; INPUT PARAMETERS:
                        0C69  1580 ;
                        0C69  1581 ;       None
                        0C69  1582 ;
                        0C69  1583 ; OUTPUTS:
                        0C69  1584 ;
                        0C69  1585 ;       R0 = STATUS CODE
                        0C69  1586 ;
                        0C69  1587 ;--
                        0C69  1588 SIP_XQP_MERGE:
                  0000  0C69  1589         .WORD 0
                        0C6B  1590         $IMGACT_S NAME = XQP_NAME,-
                        0C6B  1591                   DFLNAM = XQP_DEF,-
                        0C6B  1592                   INADR = XQP_INADDR,-
                        0C6B  1593                   IMGCTL = #IAC$M_MERGE+IAC$M_EXPREG,-
                        0C6B  1594                   RETADR = XQP_RETADDR,-
                        0C6B  1595                   HDRBUF = XQP_HEADER
              10 50   E9  0C96  1596         BLBC    R0,10$
                        0C99  1597         $IMGFIX_S
              06 50   E9  0CA0  1598         BLBC    R0,10$
00000243'FF     17    0CA3  1599         JMP     @XQP_RETADDR
                  04    0CA9  1600 10$:    RET
```

```
                                0CAA  1602          .SBTTL  SIP_MAPXQP - Create global sections for XQP
                                0CAA  1603 ;++
                                0CAA  1604 ; FUNCTIONAL DESCRIPTION:
                                0CAA  1605 ;
                                0CAA  1606 ;       Create the global sections needed to map the XQP into processes
                                0CAA  1607 ;       The SYSGEN parameter controls whether or not they are resident sections.
                                0CAA  1608 ;
                                0CAA  1609 ; INPUT PARAMETERS:
                                0CAA  1610 ;
                                0CAA  1611 ;       XQP IMAGE HEADER
                                0CAA  1612 ;
                                0CAA  1613 ; OUTPUTS:
                                0CAA  1614 ;
                                0CAA  1615 ;       R0 = STATUS CODE
                                0CAA  1616 ;
                                0CAA  1617 ;--
                                0CAA  1618
                                0CAA  1619 SIP_MAPXQP:
                        003C    0CAA  1620          .WORD   ^M<R2,R3,R4,R5>
     52    0000'CF     3C      0CAC  1621          MOVZWL  W^SIP_AERLBUFFER+IHD$W_SIZE,R2 ; OFFSET IN IMAGE HEADER TO ISD
52   00000000'8F      C0      0CB1  1622          ADDL    #SIP_AERLBUFFER,R2      ; ADDRESS OF FIRST ISD
                   62   B5   0CB8  1623 10$:     TSTW    ISD$W_SIZE(R2)          ; ARE WE DONE
                   2D   13   0CBA  1624          BEQL    25$                     ; YES
                   24   19   0CBC  1625          BLSS    20$                     ; ERROR - THERE CAN'T BE THIS MANY ISD'S
         00000000'EF   D5   0CBE  1626          TSTL    XQP$GL_DZRO             ; HAVE WE ALREADY SEEN DZRO
                   1C   12   0CC4  1627          BNEQ    20$                     ; YES - IT WAS SUP[POSED TO BE LAST
         00020405 8F   D3   0CC6  1628          BITL    #ISD$M_DZRO ! ISD$M_VECTOR ! ISD$M_GBL ! ISD$M_FIXUPVEC -
                  08 A2        0CCC  1629                  ,ISD$L_FLAGS(R2)
                   12   12   0CCE  1630          BNEQ    20$                     ; ILLEGAL ISD TYPES
      55    02 A2   3C   0CD0  1631          MOVZWL  ISD$W_PAGCNT(R2),R5     ; PAGES IN THIS SECTION
   11 08 A2      01   E1   0CD4  1632          BBC     #ISD$V_CRF,ISD$L_FLAGS(R2),30$ ; A NORMAL SECTION
00000000'EF      55   D0   0CD9  1633          MOVL    R5,XQP$GL_DZRO          ; REMEMBER HOW BIG DZRO IS
                   5E   11   0CE0  1634          BRB     50$                     ; NEXT
                        0CE2  1635 ;
            50  0000'0000'8F   D0   0CE2  1636 20$:     MOVL    #SS$_BADIMGHDR,R0
                        04   0CE9  1637 25$:     RET
                        0CEA  1638 ;
            50  0000C001 8F   D0   0CEA  1639 30$:     MOVL    #<SEC$M_GBL!SEC$M_PERM!SEC$M_SYSGBL>,R0 ; DEFAULT CHARACTERISTICS
04 00000000'EF  00000000'8F   E1   0CF1  1640          BBC     #EXE$V_XQP_RESIDENT,EXE$GL_STATIC_FLAGS,40$ ;CHECK SYSGEN PARAMETER
             00 50    0D      E2   0CFD  1641          BBSS    #SEC$V_RESIDENT,R0,40$  ; REQUEST A RESIDENT SECTION
                              0D01  1642 40$:     $CRMPSC_S -                     ; MAP A GLOBAL SECTION
                              0D01  1643                  FLAGS  = R0,-
                              0D01  1644                  GSDNAM = XQP_GSD_DESC,-
                              0D01  1645                  VBN    = ISD$L_VBN(R2),-
                              0D01  1646                  CHAN   = XQPFAB+FAB$L_STV,-
                              0D01  1647                  ACMODE = #PSL$C_EXEC,-
                              0D01  1648                  PAGCNT = R5
                BE 50    E9   0D28  1649          BLBC    R0,25$
         00000000'EF   D6   0D2B  1650          INCL    XQP$GL_SECTIONS         ; COUNT THIS SECTION
      20 00000000'EF   91   0D31  1651          CMPB    XQP$GL_SECTIONS,#32
                   A8   13   0D38  1652          BEQL    20$                     ; TOO MANY ISD'S
         000001FD'EF   96   0D3A  1653          INCB    XQP_GSDNAM+XQP_GSDNAM_SIZ-1 ; NEXT GLOBAL SECTION NAME
                              0D40  1654 ;
                53   62   3C   0D40  1655 50$:     MOVZWL  ISD$W_SIZE(R2),R3
                52   53   C0   0D43  1656          ADDL    R3,R2
                   FF6F   31   0D46  1657          BRW     10$                     ; NEXT ISD
```

```
                        0D49  1659          .SBTTL  SIP_IMAGE_ATT - Read header, get image attributes
                        0D49  1660 ;++
                        0D49  1661 ; FUNCTIONAL DESCRIPTION:
                        0D49  1662 ;
                        0D49  1663 ;       READ THE IMAGE HEADER OF AN IMAGE AND RETURN THE COUNT OF
                        0D49  1664 ;       IMAGE HEADER BLOCKS AND THE HIGHEST VBN THAT IS PART OF THE
                        0D49  1665 ;       IMAGE, I.E. EXCLUDING SYMBOL TABLE AND PATCH STUFF.
                        0D49  1666 ;
                        0D49  1667 ; INPUT PARAMETERS:
                        0D49  1668 ;
                        0D49  1669 ;       R0 = CHANNEL TO READ FROM
                        0D49  1670 ;       R1 = DISK ADDRESS TO READ (LBN OR VBN)
                        0D49  1671 ;       R2 = LAST VBN IN FILE
                        0D49  1672 ;       R3 = FUNCTION CODE (READ LOGICAL OR READ VIRTUAL)
                        0D49  1673 ;
                        0D49  1674 ; OUTPUTS:
                        0D49  1675 ;
                        0D49  1676 ;       R0 = STATUS CODE
                        0D49  1677 ;       R1 = HEADER BLOCK COUNT
                        0D49  1678 ;       R2 = LAST VIRTUAL BLOCK NUMBER IN IMAGE
                        0D49  1679 ;            EXCLUDING DEBUG SYMBOL TABLE AND PATCH AUDIT TRAIL TEXT.
                        0D49  1680 ;       R3 = IMAGE HEADER ADDRESS
                        0D49  1681 ;
                        0D49  1682 ;--
                        0D49  1683
                        0D49  1684 SIP_IMAGE_ATT:
                        0D49  1685          $QIOW_S -                       ; READ THE IMAGE HEADER
                        0D49  1686                  EFN  = #1 -             ; EVENT FLAG
                        0D49  1687                  CHAN = R0 -             ; CHANNEL TO READ ON
                        0D49  1688                  FUNC = R3 -             ; READ VIRTUAL OR LOGICAL
                        0D49  1689                  IOSB = W^SIP_Q_STATBLK - ; I/O STATUS BLOCK ADDRESS
                        0D49  1690                  P1   = W^SIP_A_ERLBUFFER - ; BUFFER TO READ INTO
                        0D49  1691                  P2   = #512 -           ; NUMBER OF BYTES TO READ
                        0D49  1692                  P3   = R1               ; DISK BLOCK TO READ
          13 50   E9    0D6E  1693          BLBC    R0,100$                 ; BRANCH IF ERROR
50  00F0'CF   3C    0D71  1694          MOVZWL  W^SIP_Q_STATBLK,R0      ; GET I/O STATUS
          0B 50   E9    0D76  1695          BLBC    R0,100$                 ; BRANCH IF ERROR
53  0000'CF   DE    0D79  1696          MOVAL   W^SIP_A_ERLBUFFER,R3    ; HEADER BUFFER ADDRESS
              0004  30    0D7E  1697          BSBW    BOC$IMAGE_ATT           ; GET IMAGE ATTRIBUTES
          50    00' D0    0D81  1698          MOVL    S^#SS$_NORMAL,R0
                    05    0D84  1699 100$:    RSB
```

```
                                                H 2
                     0D85  1702                .SBTTL  BOO$IMAGE_ATT - Get image attributes from image header
                     0D85  1703 ;++
                     0D85  1704 ; Functional Description:
                     0D85  1705 ;
                     0D85  1706 ;       BOO$IMAGE_ATT returns to the caller some attributes of the image
                     0D85  1707 ;
                     0D85  1708 ; Calling Sequence:
                     0D85  1709 ;
                     0D85  1710 ;       BSBW    BOO$IMAGE_ATT
                     0D85  1711 ;
                     0D85  1712 ; Inputs:
                     0D85  1713 ;
                     0D85  1714 ;       R2 = Size of file in blocks
                     0D85  1715 ;       R3 = Address of image header block (first one only)
                     0D85  1716 ;
                     0D85  1717 ; Outputs:
                     0D85  1718 ;
                     0D85  1719 ;       R1 = Number of image header blocks at the front of the image
                     0D85  1720 ;       R2 = Size of image in blocks excluding the blocks at the end
                     0D85  1721 ;            containing local symbols, global symbols, or patch text
                     0D85  1722 ;
                     0D85  1723 ;--
                     0D85  1724
                     0D85  1725 BOO$IMAGE_ATT::
   50   04 A3   3C   0D85  1726          MOVZWL  IHD$W_SYMDBGOFF(R3),R0  ; ANY SYMBOL TABLE INFORMATION?
          0D   13   0D89  1727          BEQL    20$                     ; BRANCH IF NOT
   51   6043   9E   0D8B  1728          MOVAB   IHS$L_DSTVBN(R0)[R3],R1 ; ADR OF 1ST VBN IN DEBUG SYMBOL TABLE
          19   10   0D8F  1729          BSBB    40$                     ; PROCESS IT
51   04 A043   9E   0D91  1730          MOVAB   IHS$L_GSTVBN(R0)[R3],R1 ; ADR OF 1ST VBN IN GLOBAL SYMBOL TABLE
          12   10   0D96  1731          BSBB    40$                     ; PROCESS IT
   50   08 A3   3C   0D98  1732 20$:    MOVZWL  IHD$W_PATCHOFF(R3),R0   ; ANY PATCH CONTROL INFORMATION?
          07   13   0D9C  1733          BEQL    30$                     ; BRANCH IF NOT
51   20 A043   9E   0D9E  1734          MOVAB   IHP$L_PATCOMTXT(R0)[R3],R1 ; ADR OF 1ST VBN OF PATCH COMMAND TEXT
          05   10   0DA3  1735          BSBB    40$                     ; PROCESS IT
   51   10 A3   9A   0DA5  1736 30$:    MOVZBL  IHD$B_HDRBLKCNT(R3),R1  ; GET IMAGE HEADER BLOCK COUNT
                05   0DA9  1737          RSB
                     0DAA  1738 ;
                     0DAA  1739 ; SEE IF VBN IS NON ZERO AND THEN IF IT IS SMALLER THAN THE CURRENT SMALLEST
                     0DAA  1740 ;
51   61   01   C3   0DAA  1741 40$:     SUBL3   #1,(R1),R1              ; FETCH VBN - 1
          08   19   0DAE  1742          BLSS    50$                     ; BRANCH IF NO VBN IS PRESENT
     51   52   D1   0DB0  1743          CMPL    R2,R1                   ; IS IT SMALLER THAN THE CURRENT ONE
          03   15   0DB3  1744          BLEQ    50$                     ; BRANCH IF NOT
     52   51   D0   0DB5  1745          MOVL    R1,R2                   ; YES, USE IT
                05   0DB8  1746 50$:    RSB
```

```
                        1749          .SBTTL  SYSTEM INITIALIZATION KERNEL LEVEL
                        1750 ;++
                        1751 ; FUNCTIONAL DESCRIPTION:
                        1752 ;
                        1753 ;       THIS ROUTINE IS CALLED TO PERFORM SYSTEM INITIALIZATION
                        1754 ;       FUNCTIONS WHICH REQUIRE KERNEL LEVEL ACCESS.
                        1755 ;       THE FOLLOWING ARE PERFORMED:
                        1756 ;
                        1757 ;               1) SET UP THE KNOWN FILE DATA BASE
                        1758 ;               2) INIT THE PAGING FILE
                        1759 ;               3) INIT THE SWAP FILE
                        1760 ;               4) MAP RMS INTO SYSTEM SPACE
                        1761 ;               6) RECOVER UNLOGGED ERROR LOG ENTRIES FROM CRASH DUMP
                        1762 ;                  AND MAKE SURE THEY ARE PROPERLY LOGGED.
                        1763 ;
                        1764 ; CALLING SEQUENCE:
                        1765 ;
                        1766 ;       ENTER VIA THE CHANGE MODE TO SYSTEM SERVICE
                        1767 ;
                        1768 ; INPUT PARAMETERS:
                        1769 ;
                        1770 ;       NONE
                        1771 ;
                        1772 ; IMPLICIT INPUTS:
                        1773 ;
                        1774 ;       LOCATION "SIP_A_FILATT" CONTAINS A LIST OF ADDRESSES OF
                        1775 ;       FILE ATTRIBUTES BUFFERS FOR:
                        1776 ;               1) PAGE FILE
                        1777 ;               2) SWAP FILE
                        1778 ;               3) RMS
                        1779 ;
                        1780 ;       THE FORMAT OF THE ATTRIBUTES BUFFERS IS:
                        1781 ;               .LONG   STARTING LBN IF CONTIGUOUS, 0 IF NOT, -1 IF NO SUCH FILE
                        1782 ;               .LONG   SIZE OF FILE IN 512 BYTE BLOCKS
                        1783 ;               .LONG   FIRST VBN IF IMAGE FORMAT
                        1784 ;               .LONG   SIZE IF IMAGE FORMAT
                        1785 ;               .LONG   BYTE COUNT OF RETRIEVAL POINTERS THAT FOLLOW
                        1786 ;               .LONG   BLOCK COUNT FOR RTRV PTR 1
                        1787 ;               .LONG   LBN FOR RTRV PTR 1
                        1788 ;               ...
                        1789 ;               ...
                        1790 ;               ...
                        1791 ;               .LONG   BLOCK COUNT FOR RTRV PTR N
                        1792 ;               .LONG   LBN FOR RTRV PTR N
                        1793 ;
                        1794 ; OUTPUT PARAMETERS:
                        1795 ;
                        1796 ;       NONE
                        1797 ;
                        1798 ; IMPLICIT OUTPUTS:
                        1799 ;
                        1800 ;       NONE
                        1801 ;
                        1802 ; COMPLETION CODES:
                        1803 ;
                        1804 ;       R0 IS RETURNED TRUE OF FALSE DEPENDING ON
                        1805 ;       INITIALIZATION SUCESS OR FAILURE
```

```
                                ;
                                ; SIDE EFFECTS:
                                ;
                                ;
                                $IP_KERNELRTN:                                  ;
                        0FFC    .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; ENTRY MASK
      53    00000000'9F   D0    MOVL    @#MMG$GL_GPTE,R3                ; GET BASE ADDRESS OF GPTE
04 A3   00000000'9F   02  78    ASHL    #2,@#SGN$GL_MAXGPGCT,4(R3); SET THE GLOBAL PAGE TABLE ENTRY
                                                                        ; MAX CNT
      50    00000000'EF   D0    MOVL    EXE$GL_SYSUCB,R0                ; PICK UP ADDRESS OF SYSTEM UCB
            01080000 8F   CA    BICL    #<DEV$M_FOR!DEV$M_MNT>,-; CLEAR FOREIGN AND MOUNTED FORM INIT IN
                 38 A0                  UCB$L_DEVCHAR(R0)               ; SYSTEM DISK UCB CHARACTERISTICS WORD
      51    00000000'EF   D0    MOVL    SCH$GL_CURPCB,R1                ; GET CURRENT PROCESS PCB ADDRESS
         2C A0    60 A1   D0    MOVL    PCB$L_PID(R1),UCB$L_PID(R0) ; ALLOCATE SYSTEM DEVICE
```

```
                                ODE6  1821          .SUBTITLE       SIP_INITPAGFIL          Initialize PAGEFILE.SYS
                                ODE6  1822 ;
                                ODE6  1823 ;  Now initialize PAGEFILE.SYS if it exists
                                ODE6  1824 ;
                                ODE6  1825
                                ODE6  1826 ;
                                ODE6  1827 ; The following register conventions are used in INITPAGFIL
                                ODE6  1828 ;
                                ODE6  1829 ;       R5 = Address of the first block of the dump header
                                ODE6  1830 ;       R6 = Address of the Boot Control Block
                                ODE6  1831 ;       R7 = Number of blocks of page file to permanently reserve for
                                ODE6  1832 ;            a dump file header.  0 if dump file is not in the page file
                                ODE6  1833 ;            4 if the dump file is in the page file.
                                ODE6  1834 ;       R8 = Number of blocks of page file to initially mark "in use"
                                ODE6  1835 ;            because the dump is in the page file and is supposed to
                                ODE6  1836 ;            be analyzed before the pages are released to the page file.
                                ODE6  1837 ;       R9 = Contents of SIP_L_PAGATT, 0 if page file contains the dump.
                                ODE6  1838 ;            The page file attributes block address if not.
                                ODE6  1839 ;
                                ODE6  1840 SIP_INITPAGFIL:
                                ODE6  1841 ;
                                ODE6  1842 ; Since the dump file may be at the front of the page file,
                                ODE6  1843 ; we will read the 3 header blocks of the dump file and
                                ODE6  1844 ; process some information now.  Later the "restore error log"
                                ODE6  1845 ; code will not have to read or update the dump.  It will only
                                ODE6  1846 ; have to process and save the error log entries if any.
                                ODE6  1847 ;
            57   7C              ODE6  1848          CLRQ    R7                      ; Init for separate dump and page files
    59  0128'CF   D0              ODE8  1849          MOVL    W^SIP_L_PAGATT,R9       ; Page file attribute block address
               03   12            ODED  1850          BNEQ    5$                      ; Branch if separate page and dump files
            57   04   D0          ODEF  1851          MOVL    #4,R7                   ; Dump is in page file
                                ODF2  1852                                          ; never page to first 4 blocks
56  00000000'EF   D0              ODF2  1853 5$:      MOVL    EXE$GL_BOOTCB,R6        ; Address of Boot Control Block
55  0000'CF   9E              ODF9  1854          MOVAB   W^SIP_A_ERLBUFFER,R5    ; Buffer to read into
          0134'CF   DD          ODFE  1855          PUSHL   W^SIP_L_DSKCHAN         ; Channel to read disk
               21   DD          0E02  1856          PUSHL   #IO$_READLBLK           ; Read function
7E   03   09   9C              0E04  1857          ROTL    #9,#3,-(SP)             ; Assume reading 3 pages
  03   1C   A6   D1              0E06  1858          CMPL    BOO$L_DMP_SIZE(R6),#3   ; Is dump file at least that big?
               02   18            0E0C  1859          BGEQ    10$                     ; Branch if yes
               6E   D4          0E0E  1860          CLRL    (SP)                    ; No blocks to be read
               65   9F          0E10  1861 10$:     PUSHAB  (R5)                    ;
            20   A6   DD          0E12  1862          PUSHL   BOO$L_DMP_MAP(R6)       ; Virtual to logical map for dump file
            18   A6   DD          0E15  1863          PUSHL   BOO$L_DMP_VBN(R6)       ; Starting VBN of dump file
               06   DD          0E18  1864          PUSHL   #6                      ; 6 arguments to RWVB
                                0E1A  1865 ;
                                0E1A  1866 ; At this point there is an argument list at the top of the stack
                                0E1A  1867 ; for the call to QIO_RWVB.  This argument list is kept until
                                0E1A  1868 ; exiting this "paragraph" when a write of the first block of the
                                0E1A  1869 ; dump header may be needed.
                                0E1A  1870 ;
          0124'CF   D4          0E1A  1871          CLRL    W^SIP_L_ERRSEQ          ; Zero saved sequence number
            10   AE   D5          0E1E  1872          TSTL    16(SP)                  ; Any blocks to read?
               6E   13          0E21  1873          BEQL    60$                     ; Branch if not
1087'CF    6E   FA              0E23  1874          CALLG   (SP),W^QIO_RWVB         ; Issue QIO Read Virtual Block
            10   AE   D4          0E28  1875          CLRL    16(SP)                  ; Init for no write of page
            63   50   E9          0E2B  1876          BLBC    R0,60$                  ; Skip if error reading file
  02   06   A5   B1              0E2E  1877          CMPW    DMP$W_DUMPVER(R5),#SIP_C_DUMPVER ; Must be known dump version
```

```
                5D   12   0E32  1878          BNEQ    60$                     ; Branch if earlier system or garbage
          50   64 A5   D2   0E34  1879          MCOML   DMP$L_SYSVER(R5),R0     ; Get complement of system version
          68 A5  50   D1   0E38  1880          CMPL    R0,DMP$L_CHECK(R5)      ; Does check match?
                 53   12   0E3C  1881          BNEQ    60$                     ; Branch if earlier system or garbage
                           0E3E  1882 ;
                           0E3E  1883 ; The dump file header looks OK, indicate that we can save error log
                           0E3E  1884 ; entries if any are present.
                           0E3E  1885 ;
          0124'CF  65   D0   0E3E  1886          MOVL    DMP$L_ERRSEQ(R5),W^SIP_L_ERRSEQ ; Save sequence number
                   07   13   0E43  1887          BEQL    20$                     ; Branch if already zero on disk
                   65   D4   0E45  1888          CLRL    DMP$L_ERRSEQ(R5)        ; Save these ERL entries only once
       10 AE    01   09   9C   0E47  1889          ROTL    #9,#1,16(SP)            ; Indicate that block is to be written
                           0E4C  1890 ;
                           0E4C  1891 ; See if the dump is in the page file and if it should be preserved
                           0E4C  1892 ;
                   59   D5   0E4C  1893 20$:      TSTL    R9                      ; Separate dump and page files?
                   4F   12   0E4E  1894          BNEQ    65$                     ; Branch if yes
    2F 00000000'EF     00'  E1   0E50  1895          BBC     S^#EXE$V_SAVEDUMP,EXE$GL_FLAGS,50$ ; Branch if not
                           0E58  1896                                          ; supposed to preserve the dump
          2A 04 A5   00   E0   0E58  1897          BBS     #DMP$V_OLDDUMP,DMP$W_FLAGS(R5),50$ ; Don't preserve dump
                           0E5D  1898                                          ; if already analyzed once.
     50   0164 C5   07   CB   0E5D  1899          BICL3   #7,DMP$L_CRASHERL+EMB$K_LENGTH+EMB$L_CR_CODE(R5),R0
                           0E63  1900                                          ; Fetch crash code, zero severity
      00000000'8F   50   D1   0E63  1901          CMPL    R0,#BUG$_OPERATOR       ; "Operator Requested Shutdown?"
                   1B   13   0E6A  1902          BEQL    50$                     ; Branch if yes, don't preserve
                           0E6C  1903 ;
                           0E6C  1904 ; Loop through the memory descriptors and calculate the number of pages
                           0E6C  1905 ; of dump to preserve.
                           0E6C  1906 ;
                           0E6C  1907          ASSUME  DMP$C_NMEMDSC EQ RPB$C_NMEMDSC
             51    08   9A   0E6C  1908          MOVZBL  #DMP$C_NMEMDSC,R1       ; Max # of memory descriptors
          52    24 A5   9E   0E6F  1909          MOVAB   DMP$L_MEMDSC(R5),R2     ; Get adr of memory descriptors
50   62      18    00   EF   0E73  1910 30$:      EXTZV   #DMP$V_PAGCNT,#DMP$S_PAGCNT,(R2),R0 ; Get page cnt for this mem
                   09   13   0E78  1911          BEQL    40$                     ; BR if no more memory descriptors used
                58   50   C0   0E7A  1912          ADDL2   R0,R8                   ; Accumulate total # of pages
                           0E7D  1913          ASSUME  DMP$C_MEMDSCSIZ EQ RPB$C_MEMDSCSIZ
                52   08   C0   0E7D  1914          ADDL2   #DMP$C_MEMDSCSIZ,R2     ; Get next memory descriptor
                   F0   51   F5   0E80  1915          SOBGTR  R1,30$                  ; Loop once for each memory descriptor
                   58   D5   0E83  1916 40$:      TSTL    R8                      ; Any dump blocks to preserve?
                   0A   14   0E85  1917          BGTR    60$                     ; Branch if yes
       10 AE    01   09   9C   0E87  1918 50$:      ROTL    #9,#1,16(SP)            ; Note that we must write the block
          00 04 A5   01   E2   0E8C  1919          BBSS    #DMP$V_EMPTY,DMP$W_FLAGS(R5),60$ ; Mark dump empty for SDA
                           0E91  1920                                          ; so it will not try to analyze
                           0E91  1921                                          ; a (partially) overwritten dump
                   59   D5   0E91  1922 60$:      TSTL    R9                      ; Address of page file attributes buffer
                   0A   12   0E93  1923          BNEQ    65$                     ; Branch if SYSINIT looked up page file
                           0E95  1924 ;
                           0E95  1925 ; Dump file is in page file.  SYSBOOT "opened" PAGEFILE.SYS and called
                           0E95  1926 ; it the dump file.  So the retrieval information and the file size
                           0E95  1927 ; are in the boot control block fields for the dump file.
                           0E95  1928 ;
             52    20 A6   D0   0E95  1929          MOVL    BOO$L_DMP_MAP(R6),R2    ; Address of page file mapping data
             54    1C A6   D0   0E99  1930          MOVL    BOO$L_DMP_SIZE(R6),R4   ; Size of page file
                   08   11   0E9D  1931          BRB     70$                     ;
             52    10 A9   DE   0E9F  1932 65$:      MOVAL   RTRVLEN(R9),R2          ; Address of page file mapping data
             54    04 A9   D0   0EA3  1933          MOVL    FILESIZE(R9),R4         ; Size of page file
          50   54    07   CB   0EA7  1934 70$:      BICL3   #7,R4,R0                ; A zero length file is also useless
```

```
                50    58    C2  0EAB  1935          SUBL    R8,R0                   ; Enough room left in page file
      000001C4  8F    50    D1  0EAE  1936          CMPL    R0,#SIP_C_MINPAGFIL     ; after reserving the dump portion
                      08    18  0EB5  1937          BGEQ    80$                     ; Branch if yes
                      58    D5  0EB7  1938          TSTL    R8                      ; No, then don't preserve the dump
                      27    13  0EB9  1939          BEQL    100$                    ; Branch if too small anyway
                      58    D4  0EBB  1940          CLRL    R8                      ; No dump data preserved
                      C8    11  0EBD  1941          BRB     50$
                    02CF    30  0EBF  1942  80$:    BSBW    SIP_INIWCB              ; Allocate and init a window control block
                                0EC2  1943  ;
                                0EC2  1944  ; Set up argument list to BOO$INITPAGFIL on the stack. Ignore returned
                                0EC2  1945  ; page file index. Default MAXVBN parameter. Use WCB address returned
                                0EC2  1946  ; by SIP_INIWCB.
                                0EC2  1947  ;
                   7E   57    7D  0EC2  1948  90$:    MOVQ    R7,-(SP)                ; Count of blocks to mark "in use"
                                0EC5  1949                                          ; Starting VBN - 1 for page file
                      7E    7C  0EC5  1950          CLRQ    -(SP)                   ; Default these two parameters
                      52    DD  0EC7  1951          PUSHL   R2                      ; Store WCB address
                      54    DD  0EC9  1952          PUSHL   R4                      ;  ... and file size
      00000000'GF     06    FB  0ECB  1953          CALLS   #6,G^BOO$INITPAGFIL     ; Allocate and initialize a PFL
                   15   50    E8  0ED2  1954          BLBS    R0,120$                 ; Go on to next step if successful
                      2D    10  0ED5  1955          BSBB    CHECK_CACHE             ; Can FIL$OPENFILE cache be deallocated?
                   E8   50    E8  0ED7  1956          BLBS    R0,90$                  ; If so, try again
                51  F28F   CF    9E  0EDA  1957          MOVAB   W^INIPAGFIL,R1          ; Otherwise, report an error message
                    0317    30  0EDF  1958          BSBW    SIP_FATAL               ;  and abort the startup sequence
                                0EE2  1959  ;
                                0EE2  1960  ; Page file does not exist, or is too small to be useful
                                0EE2  1961  ;
                51  F1D5   CF    9E  0EE2  1962  100$:   MOVAB   PAGFILERR,R1           ; Display paging file error message
                    031A    30  0EE7  1963          BSBW    SIP_SYSMSG              ;
                                0EEA  1964  ;
                                0EEA  1965  ; All exits from the init page file logic must flow through here in
                                0EEA  1966  ; order to conditionally write the first dump header block back
                                0EEA  1967  ; and unconditionally clean the argument list off the stack.
                                0EEA  1968  ;
      00000000'EF   58    D0  0EEA  1969  120$:   MOVL    R8,EXE$GL_SAVEDUMP      ; Note count of blocks reserved
                10  AE    D5  0EF1  1970          TSTL    16(SP)                  ; Write the dump file header?
                      09    13  0EF4  1971          BEQL    140$                    ; Branch if not
                14  AE    20  D0  0EF6  1972          MOVL    #IO$_WRITELBLK,20(SP)   ; Change read to write
            1087'CF   6E    FA  0EFA  1973          CALLG   (SP),W^QIO_RWVB         ; Write the block
                   5E   1C    C0  0EFF  1974  140$:   ADDL    #7*4,SP                 ; Clean argument list off stack
                      10    11  0F02  1975          BRB     SIP_INITSWPFIL
```

```
                                                N 2
                     0F04  1978          .SUBTITLE       CHECK_CACHE
                     0F04  1979
                     0F04  1980 ;+
                     0F04  1981 ; This routine checks whether there is a FIL$OPENFILE cache to be deallocated.
                     0F04  1982 ; The reason why this routine is necessary here is that the BOO$INITxxxFIL
                     0F04  1983 ; procedures cannot use the local nonpaged pool allocation routine. Those
                     0F04  1984 ; procedures are shared with SYSGEN and cannot know about such specialized
                     0F04  1985 ; items as this cache in nonpaged pool.
                     0F04  1986 ;
                     0F04  1987 ; If the cache is still allocated, it is deallocated and a success status
                     0F04  1988 ; is returned.
                     0F04  1989 ;
                     0F04  1990 ; Input Parameter:
                     0F04  1991 ;
                     0F04  1992 ;       R0 low bit clear
                     0F04  1993 ;
                     0F04  1994 ; Status Code:
                     0F04  1995 ;
                     0F04  1996 ;       R0 low bit set    => FILEREAD cache successfully deallocated
                     0F04  1997 ;
                     0F04  1998 ;       R0 low bit clear  => FILEREAD cache was already  deallocated
                     0F04  1999 ;                            (previous error stands)
                     0F04  2000 ;-
                     0F04  2001
                     0F04  2002 CHECK_CACHE:
00000000'GF    D5    0F04  2003          TSTL    G^FIL$GQ_CACHE          ; Cache still allocated?
          07    13    0F0A  2004          BEQL    10$                     ; Branch if not -- original error stands
00011D9'EF    00    FB    0F0C  2005          CALLS   #0,SIP_CACHE_DALC       ; Otherwise, deallocate the cache
                05    0F13  2006 10$:     RSB                             ;  and return to caller
```

```
                                0F14  2009          .SUBTITLE       SIP_INITSWPFIL          Initialize SWAPFILE.SYS
                                0F14  2010 ;
                                0F14  2011 ;  Now initialize SWAPFILE.SYS if it exists
                                0F14  2012 ;
                                0F14  2013
                                0F14  2014 SIP_INITSWPFIL:
          00000000'GF    25    0F14  2015          TSTW    G^SGN$GW_SWPFILES       ; If requested number of swap files is
                   3D    13    0F1A  2016          BEQL    SIP_INITRMS             ;  zero, then skip this entire section
               54  012C'CF    D0    0F1C  2017          MOVL    W^SIP_L_SWPATT,R4       ; Address of swap file attributes buffer
           50   04 A4   07    CB    0F21  2018          BICL3   #7,FILESIZE(R4),R0      ; If file is empty or does not exist
                   31    13    0F26  2019          BEQL    SIP_INITRMS             ;  then skip to the next step
       7E  00000000'GF    3C    0F28  2020          MOVZWL  G^SWP$GW_SWPINC,-(SP)   ; Get value of SWPALLOCINC parameter
                 8E    50    D1    0F2F  2021          CMPL    R0,(SP)+                ; File size must be at least as large
                   25    1F    0F32  2022          BLSSU   SIP_INITRMS             ; ... so skip to next step if too small
                52   10 A4   9E    0F34  2023          MOVAB   RTRVLEN(R4),R2          ; Address of mapping data
                     0256   30    0F38  2024          BSBW    SIP_INIWCB              ; Allocate and init a window control block
                                0F3B  2025 ;
                                0F3B  2026 ; Set up argument list to BOO$INITSWPFIL on the stack. Ignore returned
                                0F3B  2027 ; page file index. Default MAXVBN parameter. Use WCB address returned
                                0F3B  2028 ; by SIP_INIWCB.
                                0F3B  2029 ;
                      7E    7C    0F3B  2030 10$:    CLRQ    -(SP)                   ; Default last two parameters
                      52    DD    0F3D  2031          PUSHL   R2                      ; Store WCB address
                   04 A4    DD    0F3F  2032          PUSHL   FILESIZE(R4)            ;  ... and file size
       00000000'GF    04    FB    0F42  2033          CALLS   #4,G^BOO$INITSWPFIL     ; Allocate and initialize a PFL
                   0D 50    E8    0F49  2034          BLBS    R0,SIP_INITRMS          ; Go on to next step if successful
                      B6    10    0F4C  2035          BSBB    CHECK_CACHE             ; Can FIL$OPENFILE cache be deallocated?
                   EA 50    E8    0F4E  2036          BLBS    R0,10$                  ; If so, try again
               51  F218 CF   9E    0F51  2037          MOVAB   W^INIPAGFIL,R1          ; Otherwise, report an error message
                     02A0   30    0F56  2038          BSBW    SIP_FATAL               ;  and abort the startup sequence
```

```
                                0F59  2041          .SBTTL  SIP_INITRMS - Install RMS Image
                                0F59  2042 ;
                                0F59  2043 ;       INSTALL RMS IMAGE AS A PAGEABLE SYSTEM SECTION
                                0F59  2044 ;
                                0F59  2045
                                0F59  2046 SIP_INITRMS:
          54    0130'CF    D0   0F59  2047          MOVL    W^SIP_L_RMSATT,R4       ; ADDRESS OF RMS FILE ATTRIBUTES BUF
            56    0C A4    D0   0F5E  2048          MOVL    IMAGESIZE(R4),R6        ; ANY PAGES TO MAP?
                    28    15   0F62  2049          BLEQ    10$                     ; BRANCH IF NOT, ERROR
      28 00000000'EF    00'  E1   0F64  2050          BBC     S^#EXE$V_SYSPAGING,EXE$GL_FLAGS,30$ ; BRANCH IF NOT
                                0F6C  2051                                          ; PAGING SYSTEM SPACE
            52    10 A4    9E   0F6C  2052          MOVAB   RTRVLEN(R4),R2          ; ADDRESS OF MAPPING DATA
                  021E     30   0F70  2053          BSBW    SIP_INIWCB              ; MAKE A WINDOW CONTROL BLOCK
          00000000'EF     DF   0F73  2054          PUSHAL  MMG$GL_RMSBASE          ; ADDRESS TO STORE BASE OF SECTION
                    0F     DD   0F79  2055          PUSHL   S^#PRT$C_UR             ; PROTECTION FOR RMS PAGES
                    56     DD   0F7B  2056          PUSHL   R6                      ; NUMBER OF PAGES TO MAP
                 08 A4     DD   0F7D  2057          PUSHL   IMAGEVBN(R4)            ; STARTING VBN TO MAP
                    52     DD   0F80  2058          PUSHL   R2                      ; WINDOW CONTROL BLOCK ADDRESS
      00000000'EF    05     FB   0F82  2059          CALLS   #5,EXE$SYS_SECTION      ; MAP RMS AS A SYSTEM SECTION
              5B    50     E8   0F89  2060          BLBS    R0,60$                  ; BRANCH IF SUCCESSFUL
          51    F217'CF     9E   0F8C  2061 10$:     MOVAB   W^RMSMAPERR,R1          ; FAILED TO MAP RMS
                  0265     30   0F91  2062          BSBW    SIP_FATAL               ; ISSUE FATAL DIAGNOSTIC
                                0F94  2063 ;
                                0F94  2064 ; IF SYSPAGING = 0 (NOT PAGING THE PAGED PORTION OF THE SYSTEM CODE),
                                0F94  2065 ; THEN CREATE SOME (WRITABLE) ADDRESS SPACE FOR RMS AND READ IT IN.
                                0F94  2066 ; RMS WILL STILL PAGE IN THE SYSTEM WORKING SET WHICH MEANS THAT THE
                                0F94  2067 ; SYSTEM WORKING SET MUST BE LARGE IF THE PAGES ARE TO STAY IN MEMORY.
                                0F94  2068 ;
    51    00000000'EF    56   C1   0F94  2069 30$:     ADDL3   R6,BOO$GL_SPTFREL,R1    ; ALLOCATE SPT FOR RMS PAGES
          00000000'EF    51   D1   0F9C  2070          CMPL    R1,BOO$GL_SPTFREH       ; ENOUGH LEFT?
                    E7   14   0FA3  2071          BGTR    10$                     ; BRANCH IF NOT
          00000000'EF    51   D0   0FA5  2072          MOVL    R1,BOO$GL_SPTFREL       ; RECORD THE ALLOCATION
                    51   56   C2   0FAC  2073          SUBL    R6,R1                   ; FIRST SPT INDEX
                52  51    09   78   0FAF  2074          ASHL    #9,R1,R2                ; FORM SYSTEM VA OF RMS
                  00 52    1F   E2   0FB3  2075          BBSS    #VA$V_SYSTEM,R2,35$     ; OR IN THE SYSTEM BIT
                                0FB7  2076 35$:
                                0FB7  2077 ;
                                0FB7  2078 ; SET UP THE ARGUMENT LIST FOR THE READ VIRTUAL CALL
                                0FB7  2079 ;
                    0134'CF     DD   0FB7  2080          PUSHL   W^SIP_L_DSKCHAN         ; CHANNEL FOR THE I/O REQUEST
                          21    DD   0FBB  2081          PUSHL   #IO$_READLBLK           ; FUNCTION CODE
                7E  56    09    78   0FBD  2082          ASHL    #9,R6,-(SP)             ; BYTE COUNT TO READ (MAY BE > 65KB)
                          52    DD   0FC1  2083          PUSHL   R2                      ; VA TO READ INTO
                       10 A4    DF   0FC3  2084          PUSHAL  RTRVLEN(R4)             ; VIRTUAL TO LOGICAL MAP
                       08 A4    DD   0FC6  2085          PUSHL   IMAGEVBN(R4)            ; STARTING VBN IN IMAGE
                                0FC9  2086 ;
                                0FC9  2087 ; NOW FILL IN THE SPT WITH DEMAND ZERO ENTRIES
                                0FC9  2088 ;
    51    00000000'FF41    DE   0FC9  2089          MOVAL   @MMG$GL_SPTBASE[R1],R1  ; ADDRESS OF FIRST SPT ENTRY
            81    0E    18   78   0FD1  2090 40$:     ASHL    #PTE$V_PROT,S^#PRT$C_URKW,(R1)+ ; STORE THE NEXT PTE
                     F9 56    F5   0FD5  2091          SOBGTR  R6,40$                  ; LOOP THROUGH ALL PAGES
                1087'CF      06   FB   0FD8  2092          CALLS   #6,W^QIO_RWVB           ; READ RMS
                       AC 50    E9   0FDD  2093          BLBC    R0,10$                  ; BRANCH IF ERROR
          00000000'EF    52   D0   0FE0  2094          MOVL    R2,MMG$GL_RMSBASE       ; SET RMS BASE ADDRESS
00000000'EF    00000000'EF    D0   0FE7  2095 60$:     MOVL    MMG$GL_RMSBASE,CTL$GL_RMSBASE ; SET RMS BASE FOR THIS PROCESS
          51    00000000'EF    D0   0FF2  2096          MOVL    EXE$GL_SYSUCB,R1        ; GET SYSTEM DEVICE UCB ADDRESS
                       2C A1    D4   0FF9  2097          CLRL    UCB$L_PID(R1)           ; DEALLOCATE SYSTEM DEVICE
```

```
                                                D 3
                                OFFC  2100          .SBTTL  RESTORE ERROR LOG BUFFERS
                                OFFC  2101 ;
                                OFFC  2102 ;       THE FOLLOWING LOOKS AT THE FIRST 3 PAGES OF THE DUMP FILE.  IF THERE
                                OFFC  2103 ;       IS INFORMATION IN THE FILE, IT THEN LOOKS FOR ERROR LOG ENTRIES THAT
                                OFFC  2104 ;       REMAINED IN THE BUFFERS AT THE TIME OF THE CRASH.  THESE ARE REMOVED
                                OFFC  2105 ;       AND PLACED IN THE CURRENT ERROR LOG BUFFERS.  THE ERROR LOG ENTRY
                                OFFC  2106 ;       FOR THE BUG_CHECK WILL BE CONTAINED IN THE ERROR LOG BUFFER PAGES
                                OFFC  2107 ;       IN THE DUMP (PAGES 2 AND 3), IF THE DUMP WAS FOR A SYSTEM PRIOR TO
                                OFFC  2108 ;       RELEASE 2.0.  RELEASE 2.0 AND SUBSEQUENT RELEASES PLACE BUG_CHECK
                                OFFC  2109 ;       ERROR LOG IN THE FIRST PAGE OF THE DUMP FILE.  THIS WAS DONE BECAUSE
                                OFFC  2110 ;       THE ERROR LOG BUFFERS COULD BE FULL AND THE BUG_CHECK INFORMATION LOST.
                                OFFC  2111 ;
                                OFFC  2112 RESTORERL:                                   ; RESTORE ERROR LOG INFORMATION
          54    0000'CF   9E    OFFC  2113          MOVAB   W^SIP_A_ERLBUFFER,R4        ; BUFFER TO READ INTO
                0124'CF   D5    1001  2114          TSTL    W^SIP_L_ERRSEQ              ; TEST SAVED SEQUENCE NUMBER
                      03  12    1005  2115          BNEQ    10$                         ; BRANCH IF ERROR LOG ENTRIES TO SAVE
                    00A3  31    1007  2116          BRW     NOERL                       ; NO ERROR LOG ENTRIES
          55        02  D0    100A  2117 10$:       MOVL    #2,R5                       ; SET NUMBER OF ERROR LOG BUFFERS
          54    0200  C4  9E  100D  2118 20$:       MOVAB   512(R4),R4                  ; POINT TO NEXT BUFFER
             5B    01  A4  9A  1012  2119          MOVZBL  ERL$B_MSGCNT(R4),R11        ; GET COUNT OF COMPLETED MSGS IN BUFFER
                      5F  13    1016  2120          BEQL    80$                         ; NO, TRY NEXT BUFFER
                57    64  9A    1018  2121          MOVZBL  ERL$B_BUSY(R4),R7           ; GET COUNT OF INCOMPLETE MSGS IN BUFFER
                5B    57  C0    101B  2122          ADDL    R7,R11                      ; GET TOTAL # OF MESSAGES TO SCAN
             57    01F4  8F    3C  101E  2123          MOVZWL  #<512-ERL$C_LENGTH>,R7      ; SET BYTES IN BUFFER
       56    08 A4    04  A4    C3  1023  2124          SUBL3   ERL$L_NEXT(R4),ERL$L_END(R4),R6 ; EMPTY BUFFER SIZE
                57    56  D1    1029  2125          CMPL    R6,R7                       ; CHECK FOR REASONABLE POINTERS
                      49  1A    102C  2126          BGTRU   80$                         ; NO, TRY NEXT BUFFER
                57    56  C2    102E  2127          SUBL    R6,R7                       ; COMPUTE ALLOCATED SPACE IN BUFFER
                5A    0C  9A    1031  2128          MOVZBL  #ERL$C_LENGTH,R10           ; SET INITIAL OFFSET IN ERL BUFFER
             58    644A  9E    1034  2129 30$:       MOVAB   (R4)[R10],R8                ; COMPUTE MESSAGE BASE ADDRESS
                58    04  C0    1038  2130          ADDL    #EMB$K_LENGTH,R8            ; POINT PAST MESSAGE HEADER
             59    FC  A8  3C    103B  2131          MOVZWL  EMB$W_SIZE(R8),R9           ; GET MESSAGE SIZE
                      36  13    103F  2132          BEQL    80$                         ; NULL - ERROR
                57    59  C2    1041  2133          SUBL    R9,R7                       ; CHECK FOR FIT IN ALLOCATED BUFFER
                      31  19    1044  2134          BLSS    80$                         ; NO SKIP REST OF BUFFER
                   FF  A8  95    1046  2135          TSTB    EMB$B_VALID(R8)             ; IS THIS A VALID MESSAGE?
                      26  13    1049  2136          BEQL    40$                         ; BRANCH IF NOT
             01    FE  A8  91    104B  2137          CMPB    EMB$B_BUFIND(R8),#1         ; FURTHER CHECK MESSAGE VALIDITY
                      26  1A    104F  2138          BGTRU   80$                         ; BR IF NOT TO SKIP BUFFER
                59    04  C2    1051  2139          SUBL    #EMB$K_LENGTH,R9            ; SIZE OF BUFFER TO ALLOCATE
                51    59  D0    1054  2140          MOVL    R9,R1                       ; SIZE OF BUFFER TO ALLOCATE
          00000000'EF       16  1057  2141          JSB     ERL$ALLOCEMB                ; ALLOCATE BUFFER FOR MESSAGE
                   11  50  E9    105D  2142          BLBC    R0,40$                      ; SKIP IF NO SPACE
                      3F  BB    1060  2143          PUSHR   #^M<R0,R1,R2,R3,R4,R5>      ; SAVE MOVC REGISTERS
          62    68        59  28  1062  2144          MOVC3   R9,(R8),(R2)                ; COPY MESSAGE ENTIRELY
                      3F  BA    1066  2145          POPR    #^M<R0,R1,R2,R3,R4,R5>      ; RESTORE MOVC REGISTERS
          00000000'EF       16  1068  2146          JSB     ERL$RELEASEMB               ; MARK MESSAGE COMPLETE
                59    04  C0    106E  2147          ADDL    #EMB$K_LENGTH,R9            ; SIZE OF MESSAGE BUFFER W/HEADER
                5A    59  C0    1071  2148 40$:       ADDL    R9,R10                      ; POINT TO NEXT MESSAGE
                   BD  5B  F5    1074  2149 50$:       SOBGTR  R11,30$                     ; GET NEXT MESSAGE IF ANY
                   93  55  F5    1077  2150 80$:       SOBGTR  R5,20$                      ; NEXT BUFFER
          54    0000'CF   9E    107A  2151          MOVAB   W^SIP_A_ERLBUFFER,R4        ; GET ADDRESS OF FIRST PAGE FROM DUMP
            54    70  A4  9E    107F  2152          MOVAB   DMP$L_CRASHERL+EMB$K_LENGTH(R4),R4 ; GET ADR OF CRASH ERL ENTRY
            59    FC  A4  3C    1083  2153          MOVZWL  EMB$W_SIZE(R4),R9           ; GET SIZE OF ERL ENTRY
                59    04  C2    1087  2154          SUBL    #EMB$K_LENGTH,R9            ; SET SIZE OF BUFFER TO ALLOCATE
                51    59  D0    108A  2155          MOVL    R9,R1                       ; REMEMBER SIZE
          00000000'EF       16  108D  2156          JSB     ERL$ALLOCEMB                ; ALLOCATE SPACE IN ERROR LOG BUFFER
```

```
                     0E 50    E9  1093  2157          BLBC    R0,90$                          ; BR ON ERROR, NO SPACE AVAILABLE
                        3F    BB  1096  2158          PUSHR   #^M<R0,R1,R2,R3,R4,R5>          ; SAVE MOVC REGISTERS
                 62   64       59  1098  2159          MOVC3   R9,(R4),(R2)                    ; COPY ERL ENTRY INTO ERROR LOG BUFFER
                        3F    BA  109C  2160          POPR    #^M<R0,R1,R2,R3,R4,R5>          ; RESTORE MOVC REGISTERS
                00000000'EF    16  109E  2161          JSB     ERL$RELEASEMB                   ; MARK MESSAGE COMPLETE
          00000000'EF   0124'CF    D0  10A4  2162 90$:    MOVL    W^SIP_L_ERRSEQ,ERL$GL_SEQUENCE  ; RESTORE CURRENT SEQUENCE MESSAGE
                                   10AD  2163 NOERL:
                00000000'EF    16  10AD  2164          JSB     ERL$COLDSTART                   ; LOG STARTUP
                                   10B3  2165 ;
                                   10B3  2166 ; INITIALIZATION KERNEL ROUTINE COMPLETE
                                   10B3  2167 ;
                     50    01    3C  10B3  2168          MOVZWL  #1,R0                           ; GIVE SUCCESS
                                   04  10B6  2169          RET
```

```
                        1087  2172          .SBTTL  QIO_RWVB - Read or Write Virtual Block
                        1087  2173 ;++
                        1087  2174 ; Functional Description:
                        1087  2175 ;       This routine maps the specified virtual blocks to logical blocks
                        1087  2176 ;       and reads or writes the desired number of bytes to or from the
                        1087  2177 ;       specified location in memory.
                        1087  2178 ;
                        1087  2179 ; Calling sequence:
                        1087  2180 ;       CALLG   arglist,QIO_RWVB
                        1087  2181 ;
                        1087  2182 ; Inputs:
                        1087  2183 ;       QIO_RWVB_VBN(AP)    = Virtual Block Number
                        1087  2184 ;       QIO_RWVB_MAP(AP)    = Mapping info for virtual to logical mapping:
                        1087  2185 ;               # of bytes of retrieval pointers following
                        1087  2186 ;               count of LBN's in first rtrv ptr
                        1087  2187 ;               starting LBN in first rtrv ptr
                        1087  2188 ;               count of LBN's in second rtrv ptr
                        1087  2189 ;               starting LBN in second rtrv ptr
                        1087  2190 ;
                        1087  2191 ;               ...
                        1087  2192 ;
                        1087  2193 ;               count of LBN's in last rtrv ptr
                        1087  2194 ;               starting LBN in last rtrv ptr
                        1087  2195 ;       QIO_RWVB_BUF(AP)    = Buffer Address to read into
                        1087  2196 ;       QIO_RWVB_BYTCNT(AP) = Byte count to read (up to 31 bits)
                        1087  2197 ;       QIO_RWVB_FUNC(AP)   = #IO$_READLBLK or #IO$_WRITELBLK
                        1087  2198 ;       QIO_RWVB_CHAN(AP)   = Channel assigned to disk
                        1087  2199 ;
                        1087  2200 ; Outputs:
                        1087  2201 ;       R0 = Status
                        1087  2202 ;       R1 altered
                        1087  2203 ;       All other registers preserved
                        1087  2204 ;
                        1087  2205 ;--
                        1087  2206
                        1087  2207          $OFFSET 4,POSITIVE,<-
                        1087  2208          QIO_RWVB_VBN,-
                        1087  2209          QIO_RWVB_MAP,-
                        1087  2210          QIO_RWVB_BUF,-
                        1087  2211          QIO_RWVB_BYTCNT,-
                        1087  2212          QIO_RWVB_FUNC,-
                        1087  2213          QIO_RWVB_CHAN -
                        1087  2214          >
                        0004          QIO_RWVB_VBN:
                        0008          QIO_RWVB_MAP:
                        000C          QIO_RWVB_BUF:
                        0010          QIO_RWVB_BYTCNT:
                        0014          QIO_RWVB_FUNC:
                        0018          QIO_RWVB_CHAN:
                        1087  2215
                        1087  2216 QIO_RWVB:
                  0FFC  1087  2217          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                        1089  2218          ASSUME  QIO_RWVB_MAP EQ QIO_RWVB_VBN+4
53   04 AC     7D     1089  2219          MOVQ    QIO_RWVB_VBN(AP),R3     ; R3 = VBN, R4 = Map
56   0C AC     D0     108D  2220          MOVL    QIO_RWVB_BUF(AP),R6     ; R6 = Buffer address
                        10C1  2221
                        10C1  2222          ASSUME  QIO_RWVB_FUNC EQ QIO_RWVB_BYTCNT+4
```

```
                59    10 AC    7D   10C1  2223          MOVQ    QIO_RWVB_BYTCNT(AP),R9  ; R9 = byte count, R10 = function
                5B    18 AC    D0   10C5  2224          MOVL    QIO_RWVB_CHAN(AP),R11   ; R11 = RPB adr or channel
             55 84     FD 8F    78   10C9  2225          ASHL    #-3,(R4)+,R5            ; R5 = # of rtrv ptr quad words
                                     10CE  2226                                          ; R4 = adr of 1st rtrv ptr
                   50    84    7D   10CE  2227 10$:      MOVQ    (R4)+,R0                ; R0 = # of LBN's in this rtrv ptr
                                     10D1  2228                                          ; R1 = Starting LBN in this rtrv ptr
                   53    50    C2   10D1  2229          SUBL    R0,R3                   ; Is desired VBN covered
                                     10D4  2230                                          ; by this retrieval pointer?
                         05    19   10D4  2231          BLSS    20$                     ; Branch if yes
                      F5 55    F5   10D6  2232          SOBGTR  R5,10$                  ; No, get the next rtrv ptr
                         30    11   10D9  2233          BRB     60$                     ; Desired VBN beyond EOF
                   53    7340  9E   10DB  2234 20$:      MOVAB   -(R3)[R0],R3            ; R3 = R3 + R0 - 1
                                     10DF  2235                                          ; Number of blocks from the
                                     10DF  2236                                          ; beginning of this rtrv ptr
                   51    53    C0   10DF  2237          ADDL    R3,R1                   ; Adjust starting LBN
                   50    53    C2   10E2  2238          SUBL    R3,R0                   ; and LBN count
                         03    11   10E5  2239          BRB     40$
                   50    84    7D   10E7  2240 30$:      MOVQ    (R4)+,R0                ; Get the next rtrv ptr
                                     10EA  2241 ;
                                     10EA  2242 ; R0 = number of blocks that can be read in this portion
                                     10EA  2243 ; R1 = starting LBN to read from
                                     10EA  2244 ;
                         59    DD   10EA  2245 40$:      PUSHL   R9                      ; Save desired byte count
                50       50    09   78   10EC  2246          ASHL    #9,R0,R0                ; # of bytes that can be read
                         50    59   D1   10F0  2247          CMPL    R9,R0                   ; If fewer are needed
                         03    15   10F3  2248          BLEQ    50$                     ; Then read the smaller number
                         59    50   D0   10F5  2249          MOVL    R0,R9                   ; Otherwise read all we can
                         6E    59   C2   10F8  2250 50$:      SUBL    R9,(SP)                 ; Note how much is left to be read
                         58    51   D0   10FB  2251          MOVL    R1,R8                   ; Starting LBN of read request
                               11   10   10FE  2252          BSBB    QIO_RWLB                ; Read or write the file
                         59    8E   D0   1100  2253          MOVL    (SP)+,R9                ; Recover byte left to be read
                               08   15   1103  2254          BLEQ    90$                     ; Branch if all done
                            08 50   E9   1105  2255          BLBC    R0,90$                  ; Branch if read error
                            DC 55   F5   1108  2256          SOBGTR  R5,30$                  ; Get the next retrieval pointer
             50       0000'8F       3C   110B  2257 60$:      MOVZWL  #SS$_ENDOFFILE,R0       ; Indicate EOF error
                                     04   1110  2258 90$:      RET
```

```
                                                ;  3
                     1111  2261                       .SBTTL  QIO_RWLB - Read or Write Logical Block
                     1111  2262 ;++
                     1111  2263 ; Functional Description:
                     1111  2264 ;       This routine reads/writes the specified logical block numbers
                     1111  2265 ;       from/to the boot disk.
                     1111  2266 ;
                     1111  2267 ; Calling Sequence:
                     1111  2268 ;       BSBW    QIO_RWLB
                     1111  2269 ;
                     1111  2270 ; Inputs:
                     1111  2271 ;       R6  = Buffer address (updated)
                     1111  2272 ;       R8  = Logical block number (updated)
                     1111  2273 ;       R9  = Byte count to transfer (up to 31 bits)
                     1111  2274 ;       R10 = #IO$_READLBLK or #IO$_WRITELBLK
                     1111  2275 ;       R11 = Channel assigned to disk
                     1111  2276 ; Outputs:
                     1111  2277 ;       R0 = Status
                     1111  2278 ;       R1,R6-R9 altered
                     1111  2279 ;       All other registers preserved
                     1111  2280 ;--
           0000007F  1111  2281                       IOSIZE=127
                     1111  2282 QIO_RWLB:
        5E   08   C2 1111  2283                       SUBL    #8,SP                   ; Reserve an IOSB
57  FE00 8F   3C 1114  2284 10$:    MOVZWL  #IOSIZE*512,R7          ; Assume maximum transfer
     59   57   D1 1119  2285                       CMPL    R7,R9                   ; Minimize with file size
          03   15 111C  2286                       BLEQ    20$                     ; Smaller than remaining file size
     57   59   D0 111E  2287                       MOVL    R9,R7                   ; Set to remaining file size
     50   5E   D0 1121  2288 20$:    MOVL    SP,R0                   ; Address of IOSB
                  1124  2289                       $QIOW_S -
                  1124  2290                               EFN = #0 -              ; Event flag
                  1124  2291                               CHAN = R11 -            ; Channel
                  1124  2292                               FUNC = R10 -            ; Read or write logical block
                  1124  2293                               IOSB = (R0) -           ; I/O Status block address
                  1124  2294                               P1 = (R6) -             ; Buffer address
                  1124  2295                               P2 = R7 -               ; Byte count to transfer
                  1124  2296                               P3 = R8                 ; Logical block number
     08 50   E9 1141  2297                       BLBC    R0,50$                  ; Branch if error
50   6E   3C 1144  2298                       MOVZWL  (SP),R0                 ; Get completion status
     33 50   E8 1147  2299                       BLBS    R0,90$                  ; Branch if completed successfully
        20   13 114A  2300                       BEQL    70$                     ; Branch if I/O is still in progress
                  114C  2301 ;
                  114C  2302 ; Error from QI/O
                  114C  2303 ;
     0000'8F   50   B1 114C  2304 50$:    CMPW    R0,#SS$_INSFWSL         ; Insufficient working set?
           3A   12 1151  2305                       BNEQ    100$                    ; Branch if not, report error
57   57   FF 8F   78 1153  2306                       ASHL    #-1,R7,R7               ; Try again with half the byte count
57   000001FF 8F   CA 1158  2307                       BICL    #^X1FF,R7               ; Use an integral number of pages
           C0   12 115F  2308                       BNEQ    20$                     ; Branch if something left to transfer
           2A   11 1161  2309                       BRB     100$                    ; Couldn't even transfer 1 page
                  1163  2310 ;
                  1163  2311 ; The following magic with event flag 0 and the IOSB is to take care
                  1163  2312 ; of the case that the event flag was set for some reason other than
                  1163  2313 ; the completion of this particular I/O request.  In that case, the
                  1163  2314 ; only real completion information is the IOSB itself.  The sequence
                  1163  2315 ; must be to clear the event flag, check the IOSB, and then wait again
                  1163  2316 ; for the event flag.
                  1163  2317 ;
```

```
                                                1  3
                     1163  2318 60$:    $WAITFR_S #0                    ; Wait for event flag
                     116C  2319 70$:    $CLREF_S #0                     ; Clear the event flag
          50   6E   3C  1175  2320       MOVZWL  (SP),R0                 ; Fetch I/O status
               E9   13  1178  2321       BEQL    60$                     ; Branch if I/O not completed
            10 50   E9  117A  2322       BLBC    R0,100$                 ; Branch if error
                     117D  2323 ;
                     117D  2324 ; I/O completed successfully, see if there is any more to do
                     117D  2325 ;
    51    57   F7 8F   78  117D  2326 90$:    ASHL    #-9,R7,R1               ; Block count
               58   51   C0  1182  2327       ADDL    R1,R8                   ; Starting LBN for next piece
               56   57   C0  1185  2328       ADDL    R7,R6                   ; Starting Buf Adr for next piece
               59   57   C2  1188  2329       SUBL    R7,R9                   ; Count bytes tranferred
                    87   14  1188  2330       BGTR    10$                     ; Branch if another transfer to do
               5E   08   C0  118D  2331 100$: ADDL    #8,SP                   ; Clean off IOSB
                         05  1190  2332       RSB                             ; and return
```

```
                                1191  2335          .SBTTL  SIP_INIWCB - ALLOCATE AND INIT A WINDOW CONTROL BLOCK
                                1191  2336 ;
                                1191  2337 ; INPUTS:
                                1191  2338 ;
                                1191  2339 ;       R2 = ADDRESS OF MAPPING DATA
                                1191  2340 ;               # of bytes of retrieval pointers following
                                1191  2341 ;               count of LBN's in first rtrv ptr
                                1191  2342 ;               starting LBN in first rtrv ptr
                                1191  2343 ;               count of LBN's in second rtrv ptr
                                1191  2344 ;               starting LBN in second rtrv ptr
                                1191  2345 ;
                                1191  2346 ;               ...
                                1191  2347 ;
                                1191  2348 ;               count of LBN's in last rtrv ptr
                                1191  2349 ;               starting LBN in last rtrv ptr
                                1191  2350 ;
                                1191  2351 ; OUTPUTS:
                                1191  2352 ;
                                1191  2353 ;       R2 = WINDOW CONTROL BLOCK ADDRESS
                                1191  2354 ;       R0,R1,R3 ALTERED
                                1191  2355 ;       R4,R5 PRESERVED
                                1191  2356 ;       RETURNS IN LINE ONLY IF SUCCESSFUL
                                1191  2357 ;       FATAL ERROR IF FAIL TO ALLOCATE A WINDOW
                                1191  2358 ;
                                1191  2359 SIP_INIWCB:
          51    82    D0      1191  2360          MOVL    (R2)+,R1                ;SIZE OF RETRIEVAL POINTER
    53  00000000'EF    D0      1194  2361          MOVL    EXE$GL_SYSUCB,R3        ;SYSTEM UCB ADDRESS
        00000000'GF    16      119B  2362          JSB     G^FIL$INIWCB            ;ALLOCATE AND INIT THE WINDOW
              01 50    E9      11A1  2363          BLBC    R0,10$                  ;BRANCH IF FAILED
                       05      11A4  2364          RSB
                               11A5  2365
          51  F042 CF    9E   11A5  2366 10$:     MOVAB   W^INIWCBERR,R1          ; ERROR INITING WINDOW CONTROL BLOCK
                004C    30     11AA  2367          BSBW    SIP_FATAL               ;
```

```
                                11AD  2370          .SBTTL  ALLOCATE NON-PAGED DYNAMIC MEMORY
                                11AD  2371 ;++
                                11AD  2372 ; Functional Description:
                                11AD  2373 ;       This routine allocates and zeroes the specified number of
                                11AD  2374 ;       bytes of non-paged dynamic memory.  If the allocation fails
                                11AD  2375 ;       it deallocates the FIL$OPENFILE cache if has not already been
                                11AD  2376 ;       deallocated and tries again.
                                11AD  2377 ;
                                11AD  2378 ; Calling Sequence:
                                11AD  2379 ;       BSBW    SIP_ALONONPAGED
                                11AD  2380 ;
                                11AD  2381 ; Inputs:
                                11AD  2382 ;       R1 = Desired number of bytes to allocate
                                11AD  2383 ;
                                11AD  2384 ; Outputs:
                                11AD  2385 ;       R0 = Status
                                11AD  2386 ;       R1 = No. of bytes allocated if successful
                                11AD  2387 ;       R2 = Address of block allocated if successful
                                11AD  2388 ;            Block is zeroed
                                11AD  2389 ;       All other registers preserved
                                11AD  2390 ;--
                                11AD  2391
                                11AD  2392 SIP_ALONONPAGED:
                     3A   BB    11AD  2393          PUSHR   #^M<R1,R3,R4,R5>        ; REMEMBER SIZE FOR RETRY
                                11AF  2394                                          ; SAVE OTHERS FROM MOVC5
          00000000'GF   16      11AF  2395 10$:     JSB     G^EXE$ALONONPAGED       ; ALLOCATE NON-PAGED MEMORY
                11 50   E8      11B5  2396          BLBS    R0,30$                  ; BRANCH IF SUCCESSFUL
          00000000'GF   D5      11B8  2397          TSTL    G^FIL$GQ_CACHE          ; CACHE STILL ALLOCATED?
                   13   13      11BE  2398          BEQL    40$                     ; BRANCH IF NOT, ALLOC ERROR
           D9'AF   00   FB      11C0  2399          CALLS   #0,B^SIP_CACHE_DALC     ; DEALLOCATE FIL$OPENFILE CACHE
                51 6E   D0      11C4  2400          MOVL    (SP),R1                 ; RECOVER SIZE TO ALLOCATE
                   E6   11      11C7  2401          BRB     10$                     ; AND TRY AGAIN
                   07   BB      11C9  2402 30$:     PUSHR   #^M<R0,R1,R2>           ; SAVE RETURN INFO FROM MOVC5
62   51   00   62   00   2C      11CB  2403          MOVC5   #0,(R2),#0,R1,(R2)      ; ZERO THE ALLOCATED BLOCK
                   07   BA      11D1  2404          POPR    #^M<R0,R1,R2>           ; RECOVER STATUS, SIZE, ADR
                5E 04   C0      11D3  2405 40$:     ADDL    #4,SP                   ; CLEAN OFF SAVED SIZE TO ALLOCATE
                   38   BA      11D6  2406          POPR    #^M<R3,R4,R5>           ; RESTORE OTHER REGISTERS
                   05   11D8  2407          RSB
                                11D9  2408
                                11D9  2409          .SBTTL  DEALLOCATE FIL$OPENFILE CACHE
                                11D9  2410 ;
                                11D9  2411 ; KERNEL MODE ROUTINE TO DEALLOCATE THE FIL$OPENFILE CACHE
                                11D9  2412 ;
                                11D9  2413 $IP_CACHE_DALC:
                     000C       11D9  2414          .WORD   ^M<R2,R3>
          51   00000000'GF   7D      11DB  2415          MOVQ    G^FIL$GQ_CACHE,R1       ; R1 = SIZE, R2 = ADR OF CACHE
                   0F   13      11E2  2416          BEQL    10$                     ; BRANCH IF NOT PRESENT
          00000000'GF   7C      11E4  2417          CLRQ    G^FIL$GQ_CACHE          ; DISABLE THE CACHE
               50   52   D0      11EA  2418          MOVL    R2,R0
          00000000'GF   16      11ED  2419          JSB     G^EXE$DEANONPGDSIZ      ; DEALLOCATE FIL$OPENFILE CACHE
               50   00'   D0      11F3  2420 10$:     MOVL    S^#SS$_NORMAL,R0        ; INDICATE SUCCESSFUL COMPLETION
                   04   11F6  2421          RET
```

```
                     11F7  2424          .SBTTL  SIP ERROR/MESSAGE OUTPUT
                     11F7  2425 ;++
                     11F7  2426 ; FUNCTIONAL DESCRIPTION:
                     11F7  2427 ;
                     11F7  2428 ;       THIS MODULE IS CALL TO DISPLAY AN ERROR FOR THE
                     11F7  2429 ;       SYSTEM INITIALIZATION PROCESS.
                     11F7  2430 ;
                     11F7  2431 ; CALLING SEQUENCE:
                     11F7  2432 ;
                     11F7  2433 ;       BSB     SIP_FATAL               ; DISPLAY ERROR AND EXIT
                     11F7  2434 ;       BSB     SIP_SYSMSG              ; TO DISPLAY A SYSTEM ERROR AND RETURN
                     11F7  2435 ;       BSB     SIP_PUTMSG              ; TO DISPLAY AN ERROR WITH VALUE IN R0
                2F   10 11F7  2436 ;       BSB     SIP_TYPOUT              ; TYPE OUT A MESSAGE
                     11F9  2437 ;
                     11F9  2438 ; INPUT PARAMETERS:
                     11F9  2439 ;
                     11F9  2440 ; FOR SIP_FATAL AND SIP_SYSMSG:
                     11F9  2441 ;
                     11F9  2442 ;       R0 IS ERROR CODE
                     11F9  2443 ;       R1 IS ADDRESS OF COUNTED MESSAGE STRING
                     11F9  2444 ;
                     11F9  2445 ; CALL AT SIP_TYPOUT WITH:
                     11F9  2446 ;
                     11F9  2447 ;       R0 = BYTE COUNT
                     11F9  2448 ;       R1 = ADDRESS OF STRING
                     11F9  2449 ;
                     11F9  2450 ; OUTPUT PARAMETERS:
                     11F9  2451 ;
                     11F9  2452 ;       THE MESSAGE IS DISPLAYED AND AN IMAGE EXIT IS EFFECTED IF
                     11F9  2453 ;       ENTERED AT SIP_FATAL.
                     11F9  2454 ;--
                     11F9  2455 ;
                     11F9  2456 SIP_FATAL:
                50   DD 11F9  2457          PUSHL   R0                      ; SAVE ERROR
                07   10 11FB  2458          BSB     SIP_SYSMSG              ; OUTPUT MESSAGE
00000000'9F     01   FB 11FD  2459          CALLS   #1,G^SYS$EXIT           ; TAKE EXIT WITH STATUS
                         1204  2460
                         1204  2461 ;
                         1204  2462 ; ROUTINE TO PRINT MESSAGE WITH SYSTEM ERROR CODE
                         1204  2463 ;
                         1204  2464
                         1204  2465 SIP_SYSMSG:
                05   BB  1204  2466          PUSHR   #^M<R0,R2>              ; SAVE ARGUMENT AND A REGISTER
                51   DD  1206  2467          PUSHL   R1                      ; PUSH ADDRESS OF THE TEXT STRING
          52    013A'CF  DE  1208  2468          MOVAL   W^SIP_Q_LINBUF+2,R2     ; GET THE BUFFER DESCRIPTOR
                72   62  B0  120D  2469          MOVW    (R2),-(R2)              ; SET BUFFER LENGTH
                     62  7F  1210  2470          PUSHAQ  (R2)                    ; ADDRESS OF BUFFER DESCRIPTOR
                     62  3F  1212  2471          PUSHAW  (R2)                    ; PLACE TO RETURN LENGTH
                EE58 CF  9F  1214  2472          PUSHAB  W^FAOERR                ; FORMAT STRING
00000000'9F     05   FB  1218  2473          CALLS   #5,G^SYS$FAO            ; FORMAT THE MESSAGE
                50   62  3C  121F  2474          MOVZWL  (R2),R0                 ; GET LENGTH
          51    04   A2  D0  1222  2475          MOVL    4(R2),R1                ; BUFFER ADDRESS
                04   BA  1226  2476          POPR    #^M<R2>                 ; RESTORE CALLER R2
                         1228  2477                                          ; FALL INTO TYPE OUT
                         1228  2478
                         1228  2479 SIP_TYPOUT:                             ;
                03   BB  1228  2480          PUSHR   #^M<R0,R1>              ; SAVE BUFFER AND COUNT
```

```
                122A  2481          $ASSIGN_S  W^SIP_Q_TTNAME,W^SIP_L_TTCHAN ; ASSIGN A CHANNEL TO TERMINAL
34 50   E9      123B  2482          BLBC    R0,30$                          ; BR IF ERROR ASSIGNING CHANNEL
   03   BA      123E  2483          POPR    #^M<R0,R1>                      ; RESTORE COUNT AND BUFFER
                1240  2484          $QIOW_S #0,W^SIP_L_TTCHAN,-             ; EVENT FLAG 0, TERMINAL CHANNEL
                1240  2485                  #IO$_WRITEVBLK,-                ; WRITE OPERATION
                1240  2486                  -                               ; NO I/O STATUS,AST ADDRESS OR PARAMETER
                1240  2487                  (R1),R0,-                       ; BUFFER ADDRESS IN R1,R0 CONTAINS COUNT
                1240  2488                  #0,#32                          ; NULL PARAMETER PLUS CARRAIGE CONTROL
10 50   E9      125F  2489          BLBC    R0,30$                          ; BR IF ERROR WRITING TERMINAL
                1262  2490          $DASSGN_S W^SIP_L_TTCHAN                ; REMOVE TERMINAL ASSIGNMENT
01 50   E9      126E  2491          BLBC    R0,30$                          ; BR ON DEASSIGN ERROR
        05      1271  2492          RSB                                     ; RETURN TO CALLER
                1272  2493 30$:     $CMKRNL_S B^100$                        ; GET TO KERNEL MODE
                127E  2494 ;
                127E  2495 ; FATAL ERROR ROUTINE
                127E  2496 ;
          0000  127E  2497 100$:    .WORD   0                               ; ERROR ATTEMPTING OUTPUT TO TERMINAL
                1280  2498          BUG_CHECK  SYSTRMERR,FATAL              ; REPORT FATAL ERROR
```

```
                     1284  2501          .SBTTL  SIP_SETTIME  -  SET SYSTEM TIME TO CORRECT VALUE AT STARTUP
                     1284  2502 ;++
                     1284  2503 ; FUNCTIONAL DESCRIPTION:
                     1284  2504 ;
                     1284  2505 ;       THIS ROUTINE CALLS THE LOADABLE, CPU-DEPENDENT ROUTINE, EXE$INIT_TODR,
                     1284  2506 ;       TO INITIALIZE THE TIME-OF-DAY REGISTER AND SYSTEM TIME.
                     1284  2507 ;
                     1284  2508 ; INPUT PARAMETERS:
                     1284  2509 ;
                     1284  2510 ;       NONE
                     1284  2511 ;
                     1284  2512 ; IMPLICIT INPUTS:
                     1284  2513 ;
                     1284  2514 ;       TIME-OF-DAY PROCESSOR CLOCK.
                     1284  2515 ;
                     1284  2516 ; OUTPUT PARAMETERS:
                     1284  2517 ;
                     1284  2518 ;       R0,R1 - DESTROYED
                     1284  2519 ;
                     1284  2520 ; IMPLICIT OUTPUTS:
                     1284  2521 ;
                     1284  2522 ;       EXE$GQ_SYSTIME - SET TO CURRENT TIME IN 100 NANOSECOND UNITS SINCE
                     1284  2523 ;                        17-NOV-1858  00:00:00.
                     1284  2524 ;
                     1284  2525 ;--
                     1284  2526
                     1284  2527 SIP_SETTIME:                                ; SET CORRECT TIME
                0000 1284  2528          .WORD   0                          ; ENTRY MASK
00000000'EF    16    1286  2529          JSB     EXE$INIT_TODR              ; CALL CPU-DEPENDENT ROUTINE
    50  00'  3C    128C  2530          MOVZWL  S^#SS$_NORMAL,R0           ; INDICATE SUCCESS
             04    128F  2531          RET                                ;
                     1290  2532
                     1290  2533          .END    SIP_START
```

```
$$.TAB                          =  0000007C R      04
$$.TABEND                       =  000000CC R      04
$$.TMP                          =  00000000
$$.TMP1                         =  00000001
$$.TMP2                         =  000000CF
$$T1                            =  00000001
ACPINIERR                          0000010F R      02
ATR$C_ASCNAME                   =  00000010
ATR$S_ASCNAME                   =  00000056
BOO$GL_SPTFREH                     ********   X    02
BOO$GL_SPTFREL                     ********   X    02
BOO$IMAGE_ATT                      00000D85 RG     02
BOO$INITPAGFIL                     ********   X    02
BOO$INITSWPFIL                     ********   X    02
BOO$L_DMP_MAP                   =  00000020
BOO$L_DMP_SIZE                  =  0000001C
BOO$L_DMP_VBN                   =  00000018
BUG$_OPERATOR                      ********   X    02
BUG$_SYSTRMERR                     ********   X    02
CCB$L_UCB                       =  00000000
CHECK_CACHE                        00000F04 R      02
CLU$GB_QDISK                       ********   X    04
CLU$GL_CLUB                        ********   X    02
CLUB$L_CLUDCB                   =  000000B4
CLUB$L_FLAGS                    =  0000001C
CLUB$M_INIT                     =  00080000
CLUB$Q_NEWTIME                  =  00000094
CLUB$Q_NEWTIME_REF              =  0000009C
CLUB$T_QDNAME                   =  000000B8
CLUB$V_CLUSTER                  =  00000C00
CLUDCB$L_QFLBN                  =  0000001C
CLUDCB$L_TQE                    =  00000014
CLUDCB$L_UCB                    =  0000000C
CLUDCB$M_QS_READY               =  00000002
CLUDCB$S_DISK_QUORUM            =  00000010
CLUDCB$W_STATE                  =  00000020
CMNSYS                             00000309 R      02
CNX$DISK_CHANGE                    ********   X    02
CREERREND                          000001EC R      04
CRELNMERR                          00000095 R      02
CRELNM_DONE                        0000063A R      02
CRELNM_FATAL                       00000632 R      02
CREPRCERR                          000001C4 R      04
CREPRCNAM                          000001DD R      04
CTL$GL_CCBBASE                     ********   X    02
CTL$GL_PCB                         ********   X    02
CTL$GL_RMSBASE                     ********   X    02
DEV$M_CLU                       =  00000001
DEV$M_FOR                       =  01000000
DEV$M_MNT                       =  00080000
DIR...                          =  00000001
DMP$C_MEMDSCSIZ                 =  00000008
DMP$C_NMEMDSC                   =  00000008
DMP$L_CHECK                     =  00000068
DMP$L_CRASHERL                  =  0000006C
DMP$L_ERRSEQ                    =  00000000
DMP$L_MEMDSC                    =  00000024
DMP$L_SYSVER                    =  00000064
DMP$S_PAGCNT                    =  00000018
DMP$V_EMPTY                     =  00000001
DMP$V_OLDDUMP                   =  00000000
DMP$V_PAGCNT                    =  00000000
DMP$W_DUMPVER                   =  00000006
DMP$W_FLAGS                     =  00000004
DSC$K_CLASS_S                      ********   X    04
DSC$K_DTYPE_T                      ********   X    04
DVI$_FULLDEVNAM                 =  000000E8
EMB$B_BUFIND                    =  FFFFFFFE
EMB$B_VALID                     =  FFFFFFFF
EMB$K_LENGTH                    =  00000004
EMB$L_CR_CODE                   =  000000F4
EMB$W_SIZE                      =  FFFFFFFC
ERL$ALLOCEMB                       ********   X    02
ERL$B_BUSY                      =  00000000
ERL$B_MSGCNT                    =  00000001
ERL$COLDSTART                      ********   X    02
ERL$C_LENGTH                    =  0000000C
ERL$GL_SEQUENCE                    ********   X    02
ERL$L_END                       =  00000008
ERL$L_NEXT                      =  00000004
ERL$RELEASEMB                      ********   X    02
ERROR                              0000C      R    02
EXE$ALONONPAGED                    ********   X    02
EXE$DEANONPGDSIZ                   ********   X    02
EXE$GL_BOOTCB                      ********        02
EXE$GL_FLAGS                       ********        02
EXE$GL_SAVEDUMP                    ********        02
EXE$GL_STATIC_FLAGS                ********        02
EXE$GL_SYSID_LOCK                  ********   X    02
EXE$GL_SYSMSG                      ********   X    02
EXE$GL_SYSUCB                      ********   X    02
EXE$GQ_SYSTIME                     ********   X    02
EXE$GT_STARTUP                     ********   X    04
EXE$INIT_TODR                      ********   X    02
EXE$INSTIMQ                        ********   X    02
EXE$SETIME_INT                     ********   X    02
EXE$SYS_SECTION                    ********   X    02
EXE$V_PAGFILDMP                    ********   X    02
EXE$V_SAVEDUMP                     ********   X    02
EXE$V_SYSPAGING                    ********   X    02
EXE$V_SYSUAFALT                    ********   X    02
EXE$V_XQP_RESIDENT                 ********   X    02
EXEC_MODE                          0000042B R      02
FAB$C_BID                       =  00000003
FAB$C_BLN                       =  00000050
FAB$C_FIX                       =  00000001
FAB$C_SEQ                       =  00000000
FAB$L_ALQ                       =  00000010
FAB$L_FOP                       =  00000004
FAB$L_STV                       =  0000000C
FAB$V_CHAN_MODE                 =  00000002
FAB$V_FILE_MODE                 =  00000004
FAB$V_GET                       =  00000001
FAB$V_LNM_MODE                  =  00000000
```

```
FAB$V_UFO                       = 00000011
FAB$W_GBC                       = 00000048
FAOERR                            00000070 R      02
FID$C_MFD                       = 00000004
FIL$GQ_CACHE                      ********   X    02
FIL$GT_TOPSYS                     ********   X    02
FIL$INIWCB                        ********   X    02
FIL$OPENFILE                      ********   X    02
FIL$OPENFILE_1                    ********   X    02
FILELBN                           00000000
FILESIZE                          00000004
FILOPNERR                         000001D8 R      02
IAC$M_EXPREG                    = 00000020
IAC$M_MERGE                     = 00000010
IHD$B_HDRBLKCNT                 = 00000010
IHD$W_PATCHOFF                  = 00000008
IHD$W_SIZE                      = 00000000
IHD$W_SYMDBGOFF                 = 00000004
IHP$L_PATCOMTXT                 = 00000020
IHS$L_DSTVBN                    = 00000000
IHS$L_GSTVBN                    = 00000004
IMAGESIZE                         0000000C
IMAGEVBN                          00000008
INIKNOWNFIL                       00000286 R      02
INIPAGFIL                         0000016D R      02
INIWCBERR                         000001EB R      02
IO$_ACCESS                      = 00000032
IO$_PACKACK                     = 00000008
IO$_READLBLK                    = 00000021
IO$_READVBLK                    = 00000031
IO$_WRITELBLK                   = 00000020
IO$_WRITEVBLK                   = 00000030
IOC$LOCK_DEV                      ********   X    02
IOSIZE                          = 0000007F
ISD$L_FLAGS                     = 00000008
ISD$L_VBN                       = 0000000C
ISD$M_DZRO                      = 00000004
ISD$M_FIXUPVEC                  = 00000400
ISD$M_GBL                       = 00000001
ISD$M_VECTOR                    = 00020000
ISD$V_CRF                       = 00000001
ISD$W_PAGCNT                    = 00000002
ISD$W_SIZE                      = 00000000
LCK$GB_STALLREQS                  ********   X    02
LCK$K_CRMODE                    = 00000001
LCK$K_EXMODE                    = 00000005
LCK$M_CVTSYS                    = 00000040
LCK$M_NOQUEUE                   = 00000004
LCK$M_SYNCSTS                   = 00000008
LCK$M_SYSTEM                    = 00000010
LNM$M_CONCEALED                 = 00000100
LNM$M_TERMINAL                  = 00000200
LNM$_ATTRIBUTES                 = 00000003
LNM$_STRING                     = 00000002
LNM_FILE_DEV                      00000315 R      02
LNM_SYSTEM_DESC                   00000329 R      02
LOCKDOWN                          ********   X    02
LOCKERR                           00000148 R      02
LOCK_FLAGS                      = 0000005C
LOCK_ID                           0000044F R      04
LOCK_NAME                         00000453 R      04
LOCK_NAME_DESC                    00000463 R      04
LOCK_NAME_SIZE                  = 00000010
LOCK_STATUS                       0000044B R      04
LOCK_STATUS_BLOCK                 0000044B R      04
MMG$GL_GPTE                       ********     X  02
MMG$GL_RMSBASE                    ********     X  02
MMG$GL_SPTBASE                    ********     X  02
MOUERR                            0000012B R      02
MOUNT_SYSTEM                      ********     X  02
MSGFICERR                         000000D9 R      02
MSGFILFAB                         00000000 R      04
MSGFILNAM                         000002CD R      02
MSGFILNAMSZ                     = 00000016
MSGFILXAB                         00000050 R      04
NOERL                             000010AD R      02
NO_ATTR                           00000433 R      02
PAGFILERR                         000000BB R      02
PAGFILNAM                         000002AB R      02
PCB$L_PID                       = 00000060
PQL$AB_SYSPQL                     ********     X  02
PQL$_ASTLM                      = 00000001
PQL$_BIOLM                      = 00000002
PQL$_BYTLM                      = 00000003
PQL$_CPULM                      = 00000004
PQL$_DIOLM                      = 00000005
PQL$_ENQLM                      = 0000000C
PQL$_FILLM                      = 00000006
PQL$_JTQUOTA                    = 0000000E
PQL$_LISTEND                    = 00000000
PQL$_PGFLQUOTA                  = 00000007
PQL$_PRCLM                      = 00000008
PQL$_TQELM                      = 00000009
PQL$_WSDEFAULT                  = 0000000B
PQL$_WSQUOTA                    = 0000000A
PR$_IPL                         = 00000012
PRT$C_UR                        = 0000000F
PRT$C_URKW                      = 0000000E
PSL$C_EXEC                      = 00000001
PTE$V_PROT                      = 0000001B
QIO_RQLB                          00001111 R      02
QIO_RWVB                          000010B7 R      02
QIO_RWVB_BUF                      0000000C
QIO_RWVB_BYTCNT                   00000010
QIO_RWVB_CHAN                     00000018
QIO_RWVB_FUNC                     00000014
QIO_RWVB_MAP                      00000008
QIO_RWVB_VBN                      00000004
RESTORERC                         00000FFC R      02
RMSFILNAM                         000002C5 R      02
RMSMAPERR                         000001A7 R      02
RPB$C_MEMDSCSIZ                 = 00000008
RPB$C_NMEMDSC                   = 00000008
RTRVLEN                           00000010
```

```
RTRVPTRS                        00000014
SAVABS...                     = 0000001C
SCH$GL_CURPCB                   ********    X   02
SCH$IOLOCKW                     ********    X   02
SCH$IOUNLOCK                    ********    X   02
SCS$GB_SYSTEMID                 ********    X   02
SCS$GB_SYSTEMIDH                ********    X   02
SEC$M_GBL                     = 00000001
SEC$M_PERM                    = 00004000
SEC$M_SYSGBL                  = 00008000
SEC$V_RESIDENT                = 0000000D
SGN$GC_MAXGPGCT                 ********    X   02
SGN$GW_SWPFILES                 ********    X   02
SIP_ALONONPAGED                 000011AD R      02
SIP_A_ATRLIST                   00000014 R      02
SIP_A_ERLBUFFER                 00000000 R      03
SIP_A_FIB                       000000CC R      04
SIP_A_FILATT                    00000128 R      04
SIP_A_FILEHDR                 = 00000200 R      03
SIP_A_INDEXFHDR               = 00000000 R      03
SIP_A_NAMES                     000002E3 R      02
SIP_A_OPENARG                   00000104 R      04
SIP_CACHE_DALC                  000011D9 R      02
SIP_CLUSTER_INIT                00000A04 R      02
SIP_CLU_MSG                     00000263 R      02
SIP_CLU_TIMOUT                  000004E9 R      04
SIP_C_DUMPVER                 = 00000002
SIP_C_FIB_SIZE                = 00000010
SIP_C_LINBUFSIZ               = 00000084
SIP_C_MINPAGFIL               = 000001C4
SIP_FATAL                       000011F9 R      02
SIP_GET_SYSID_LOCK              0000099E R      02
SIP_GET_TOPSYS                  000007A7 R      02
SIP_IMAGE_ATT                   00000D49 R      02
SIP_INITPAGFIL                  00000DE6 R      02
SIP_INITRMS                     00000F59 R      02
SIP_INITSWPFIL                  00000F14 R      02
SIP_INIWCB                      00001191 R      02
SIP_KERNELRTN                   00000B89 R      02
SIP_LOOKUP_QFILE                00000825 R      02
SIP_L_DSKCHAN                   00000134 R      04
SIP_L_ERRSEQ                    00000124 R      04
SIP_L_PAGATT                    00000128 R      04
SIP_L_RMSATT                    00000130 R      04
SIP_L_RTRVLEN                   00000100 R      04
SIP_L_SWPATT                    0000012C R      04
SIP_L_TTCHAN                    000000DC R      04
SIP_MAPXQP                      00000CAA R      02
SIP_QD_CHAN                     000004F1 R      04
SIP_QD_DESCR                    000004FD R      04
SIP_QD_IOSB                     000004F5 R      04
SIP_QD_ITMLST                   00000575 R      04
SIP_QD_STATBUF                  000004F5 R      04
SIP_QF_BUFFER                   00000521 R      04
SIP_QF_DESCR                    00000505 R      04
SIP_QF_NAME                     0000050D R      04
SIP_QF_NAME_SIZE              = 00000014

SIP_Q_FIBDESC                   0000000C R      02
SIP_Q_LINBUF                    00000138 R      04
SIP_Q_PRVMSK                    00000068 R      02
SIP_Q_RETADR                    000000E0 R      04
SIP_Q_RTRVBUF                   000000F8 R      04
SIP_Q_SPIMAGE                   00000049 R      02
SIP_Q_SPINPUT                   000001EC R      04
SIP_Q_SPOUTPUT                  0000002F R      02
SIP_Q_SPOUTXDT                  0000003C R      02
SIP_Q_STARTUP                   00000020 R      02
SIP_Q_STATBLK                   000000F0 R      04
SIP_Q_TMPDESC                   000000E8 R      04
SIP_Q_TTNAME                    00000000 R      02
SIP_SETTIME                     00001284 R      02
SIP_START                       00000477 R      02
SIP_START_QUORUM_TIMER          00000C45 R      02
SIP_SYSMSG                      00001204 R      02
SIP_TYPOUT                      00001228 R      02
SIP_T_LINBUF                    00000140 R      04
SIP_XQP_MERGE                   00000C69 R      02
SS$_BADIMGHDR                   ********      X 02
SS$_ENDOFFILE                   ********      X 02
SS$_INSFWSL                     ********      X 02
SS$_NORMAL                      ********      X 02
SS$_NOSUCHFILE                  ********      X 02
STAC_IMAGE                      0000046B R      04
STAC_OPER                       00000480 R      04
STAC_PRC                        00000496 R      04
STAC_PRV_MSK                    0000048E R      04
STAC_QLIST                      000004A7 R      04
STATBLK                         00000000
SWP$GW_SWPINC                   ********    X   02
SWPFILNAM                       000002B8 R      02
SYS$ASSIGN                      ********  GX    02
SYS$CLREF                       ********  GX    02
SYS$CMEXEC                      ********  GX    02
SYS$CMKRNL                      ********  GX    02
SYS$CRELNM                      ********  GX    02
SYS$CREPRC                      ********  GX    02
SYS$CRMPSC                      ********  GX    02
SYS$DASSGN                      ********  GX    02
SYS$ENQW                        ********  GX    02
SYS$EXIT                        ********   X    02
SYS$EXPREG                      ********  GX    02
SYS$FAO                         ********   X    02
SYS$GETDVIW                     ********  GX    02
SYS$GETMSG                      ********  GX    02
SYS$IMGACT                      ********  GX    02
SYS$IMGFIX                      ********  GX    02
SYS$OPEN                        ********  GX    02
SYS$QIOW                        ********  GX    02
SYS$SETIME                      ********  GX    02
SYS$SETIMR                      ********  GX    02
SYS$TRNLNM                      ********  GX    02
SYS$WAITFR                      ********  GX    02
SYSID_LOCK_ERR                  00000236 R      02
SYSUAFALT                       00000414 R      02
```

```
SYSUAFALT_LEN                   = 00000009
SYSUAF_DESC                       0000041D R      02
SYSUAF_ITMLST                     00000467 R      02
SYS_COMMON                        0000033B R      02
SYS_COMMON_DESC                   00000346 R      02
SYS_COMMON_ITMLST                 00000585 R      04
SYS_COMMON_LENGTH               = 0000000B
SYS_DISK_DESC                     000003B9 R      02
SYS_ID                            0000045D R      04
SYS_MESSAGE                       00000358 R      02
SYS_MESSAGE_DESC                  0000036C R      02
SYS_MESSAGE_ITMLST                00000437 R      02
SYS_MESSAGE_LEN                 = 00000014
SYS_SHARE                         0000037F R      02
SYS_SHARE_DESC                    00000393 R      02
SYS_SHARE_ITMLST                  00000447 R      02
SYS_SHARE_LEN                   = 00000014
SYS_SYSDEVICE_ATTR                000005BD R      04
SYS_SYSDEVICE_DESC                000003A4 R      02
SYS_SYSDEVICE_DEV                 000005B1 R      04
SYS_SYSDEVICE_DEV_LEN             000005AD R      04
SYS_SYSDEVICE_DVI_LST             000005C1 R      04
SYS_SYSDEVICE_ITMLST              000005A1 R      04
SYS_SYSROOT_CMNSYS                00000595 R      04
SYS_SYSROOT_CMNSYS_LEN            00000591 R      04
SYS_SYSROOT_DESC                  000003C9 R      02
SYS_SYSROOT_ITMLST                000005D1 R      04
SYS_SYSROOT_TOPSYS                000005E1 R      04
SYS_SYSROOT_TOPSYS_LEN            000005DD R      04
SYS_SYSTEM                        000003DC R      02
SYS_SYSTEM_DESC                   000003F0 R      02
SYS_SYSTEM_ITMLST                 00000457 R      02
SYS_SYSTEM_LEN                  = 00000014
SYS_TOPSYS_DESC                   00000402 R      02
SYS_TOPSYS_DIRNAM                 00000609 R      04
SYS_TOPSYS_DIRNAM_LEN             00000605 R      04
SYS_TOPSYS_ITMLST                 00000605 R      04
TERMINAL_CONCEALED_ATTR           0000042F R      02
TQE$L_TQFL                      = 00000000
UCB$L_DEVCHAR                   = 00000038
UCB$L_DEVCHAR2                  = 0000003C
UCB$L_PID                       = 0000002C
UCB$L_STS                       = 00000064
UCB$V_VALID                     = 0000000B
UCB$W_REFC                      = 0000005C
VA$V_SYSTEM                     = 0000001F
XAB$C_FHC                       = 0000001D
XAB$C_FHCLEN                    = 0000002C
XAB$L_EBK                       = 00000010
XAB$L_NXT                       = 00000004
XAB$W_FFB                       = 00000014
XDT$START                         ********  GX    00
XQP$GL_DZRO                       ********    X   02
XQP$GL_SECTIONS                   ********    X   02
XQPERR                            00000215 R      02
XQPFAB                            0000007C R      04
XQPNAM                            000002F5 R      02
XQPNAMSIZ                       = 00000016
XQP_DEF                           00000224 R      04
XQP_GSDNAM                        000001F4 R      04
XQP_GSDNAM_SIZ                  = 0000000A
XQP_GSD_DESC                      000001FE R      04
XQP_HEADER                        0000024B R      04
XQP_INADDR                        0000023B R      04
XQP_NAME                          00000206 R      04
XQP_RETADDR                       00000243 R      04
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	0000001C (28.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
SIP_PURE	00001290 (4752.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE
SIP_RWDATA_PAGE	00000600 (1536.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC PAGE
SIP_RWDATA	00000615 (1557.)	04 (4.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.11	00:00:00.66
Command processing	140	00:00:00.72	00:00:04.10
Pass 1	819	00:00:39.67	00:01:55.96
Symbol table sort	0	00:00:05.04	00:00:08.74
Pass 2	417	00:00:09.30	00:00:21.37
Symbol table output	1	00:00:00.39	00:00:01.02
Psect synopsis output	0	00:00:00.03	00:00:00.05
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1408	00:00:55.26	00:02:31.91

The working set limit was 2550 pages.
213249 bytes (417 pages) of virtual memory were used to buffer the intermediate code.
There were 180 pages of symbol table space allocated to hold 3236 non-local and 115 local symbols.
2533 source lines were read in Pass 1, producing 38 object records in Pass 2.
97 pages of virtual memory were used to define 90 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	28
_255DUA28:[SYSLIB]STARLET.MLB;2	55
TOTALS (all libraries)	83

3648 GETS were required to define 83 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SYSINIT/OBJ=OBJ$:SYSINIT MSRC$:SYSINIT/UPDATE=(ENH$:SYSINIT)+EXECML$/LIB

WRTMFYPAG
LIS

UCBCREDEL
LIS

USRVECTOR
LIS

SYSVECTOR
LIS

SYSINIT
LIS

VERSION
LIS

SYSINI

UPCASEDAT
LIS

SYSINIT
MAP

SYSWAIT
LIS

TIMESCHDL
LIS

0390 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SYSLOA780-
MAP

SYSLOA790
MAP

SCSLOA
MAP

SYSMOU
LIS

SYSLOAUV1
MAP

SYSLOAWS1
MAP

CSP
MAP

790DEF
MDL

SYSLOA

CLUSTRLOA
MAP

SYSLOA730
MAP

SYSLOA750
MAP